Entropy analysis of physical unclonable functions

van den Berg, R.

Award date:
2012

Entropy analysis of Physical Unclonable Functions

Robbert van den Berg

Department of Mathematics and Computer Science
Eindhoven University of Technology

Graduation supervisor:
Boris Škorić

Graduation tutor:
Vincent van der Leest

Committee:
Boris Škorić
Berry Schoenmakers
Vincent van der Leest

August 2012

Abstract

Physical Unclonable Functions (PUFs) are an emerging security primitive useful for secure key storage mechanisms and anti-counterfeiting measures. To do this securely, it is imperative that PUFs are unique, i.e., possess enough entropy.

The aim of this thesis is to develop and implement new methodologies to accurately estimate the entropy of PUFs. To this end, a novel method is presented which estimates the extractable entropy by calculating the mutual information between enrollment and reconstruction measurements. Furthermore, a method to determine uniqueness from the field of biometrics is modified in such a way that the entropy of PUFs can be estimated.

Our newly developed entropy estimation methods are compared to other methods found in the literature using a comprehensive experimental evaluation on synthetically generated PUF data with known entropy. We find that the results from our methods are in line with results found in the literature using methods such as the context-tree weighting compression and inter-device distance tests.

The contributions of our methods are that the influence of the reconstruction condition (noise) is included in the entropy estimation as well as providing insight on how entropy develops if more measurements are used during enrollment and reconstruction. We find that the extractable entropy increases as more measurements are used during enrollment and reconstruction, but only if PUF biases are reliable across conditions.

Furthermore, an entropy analysis is performed on PUF data from the European Union sponsored UNIQUE project, which consists of 192 application-specific integrated circuits housing multiple instances of the most popular intrinsic PUF types. In particular, we performed a comprehensive uniqueness evaluation of four memory based PUF types: the latch, D flip-flop (DFF), buskeeper and SRAM PUF. Our analysis shows that from the four analysed PUF types, the SRAM PUF has the most extractable entropy. The buskeeper also shows strong PUF behaviour, while the DFF and latch PUF have the least extractable entropy.

Acknowledgements

I thank several people for their help and support during, and leading up to, my graduation project. First of all, I thank my supervisor Boris Škorić, for his weekly guidance, his clear explanations and his useful feedback. I also want to thank Intrinsic-ID for giving me the opportunity to work with real PUF data and use their resources. Furthermore, I thank Vincent van der Leest at Intrinsic-ID for proofreading my thesis and providing me with useful suggestions. I also thank the people involved in the Kerckhoffs institute for a unique and very valuable master track. In particular, I thank the professors and fellow students for their willingness to share information and their efforts to make this world a safer place. Lastly, I thank my parents and brother for their moral support throughout my entire study period.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This introduction starts with the motivation for the thesis, describing why it is important to be able to reliably estimate the entropy of a physical unclonable function. Then, the research goals and contributions of this thesis are stated. Lastly, the outline of the thesis is presented.

1.1 Motivation

Anti-counterfeiting schemes and secure key storage are challenges commonly faced by the electronics industry. A promising solution is already actively researched and used, in the form of physical unclonable functions. Physical unclonable functions (PUFs) exploit irreproducible physical features to deduce a unique identifier or secret, functioning as an electronic fingerprint or secure key storage mechanism. Their physical nature makes them more resilient against cloning attacks than storing a unique identifier or key in digital form, which directly opposes the unclonability requirement.

However, it is imperative that PUFs provide enough uniqueness such that they are hard to predict. More specifically, it should hold that, given a set of PUFs, the response(s) of a specific PUF can only be predicted with negligible probability, even knowing the responses of all other PUFs.

An eminent property of a PUF is its entropy, which is a measure of uniqueness. The higher the entropy the smaller and thus cheaper a PUF can be while retaining the same level of security.

Hence, one of the key questions is determining the amount of entropy present in different PUF implementations. The aim of this thesis is to develop and implement methodologies to quickly and accurately assess the entropy of a PUF.

Using these methodologies, the entropy of both synthetic data and actual PUF data of four different PUF types will be estimated. The results will be cross-referenced with current entropy estimations in the literature. This will achieve two goals: the performance of various methods can be assessed and the entropy of different PUF types can be compared.

The actual PUF data is obtained from the European Union sponsored UNIQUE-project, in which a collaboration of several companies and universities developed an application-specific integrated circuit (ASIC) containing six different PUF types. The UNIQUE project [1] is a possibility to perform the first, to the knowledge of the author, comparison of different PUF types where each type is present in a large quantity.

1.2 Research goals

The main goals of this thesis are two-fold, comprising a theoretical and practical part. The theoretical part focusses on determining how PUF entropy can be reliably estimated. This involves the creation of a framework in which entropy estimation methods can be analysed. Using this framework, both methods found through a literature study and own methodologies are assessed using synthetically generated PUF data which enables a fair comparison.

In the practical part, these entropy estimation methods are applied to real PUF data. The resulting entropy estimations are compared across PUF types and cross-referenced with results found in the literature.

Both the theoretical and the practical research goals are captured in the following list:

1. Model the PUF creation process and create a framework suitable for analysing PUF entropy using various methods.

   The first goal is to establish a proper framework in which the PUF creation process is detailed, quantifying the relevant stochastic variables and denoting the important entropies.

2. Determine a method to accurately estimate entropy.

   To discern the uniqueness, a security quality of a PUF, it is important to determine how much information can be extracted from a PUF. The second theoretical goal is to find a method to accurately estimate the entropy of a PUF.

3. Assess the performance of various PUF entropy estimation methods using synthetic data.

   The third goal is to determine the quality of the newly introduced entropy estimation methods and compare them to methods found in the literature. In order to determine how accurate entropy estimation methods are, we apply them to synthetic data of which the true entropy is known.

4. Apply entropy estimation methods to actual PUF data.

   The practical part consists of an entropy analysis of actual PUF data. The entropy estimation methods are applied to PUF data obtained from the UNIQUE project to determine which PUF type contains the most extractable entropy per surface area.

1.3 Thesis overview

This thesis is organised in the following manner. Chapter 2 contains the preliminaries, explaining the notation used in this thesis and some definitions. Chapter 3 will introduce the concept of physical unclonable functions in more detail. First, properties of PUFs are explained and an overview of typical applications is given, as well as a brief overview of possible attacks. Then the concepts of reliability and security with respect to PUFs are explained, followed by a section on possible attacks. This chapter concludes with a discussion of all the PUF types present on the UNIQUE ASIC.

In Chapter 4, a literature study is performed discussing previous work regarding entropy estimation. The focus will mainly lie on the theoretic aspects of entropy calculation methods; the assessment of entropy evaluation methods is done in a later chapter as the results in the literature are difficult to compare due to varying testing conditions.

The purpose of Chapter 5 is to provide a theoretic framework to accurately determine PUF entropy. This framework is built from the ground up modelling the PUF creation process, detailing the relevant variables, entropies and the influence of noise.

Chapter 6 introduces new methodologies to estimate PUF entropy based on the new framework. First, two methods are provided to estimate how the biases of PUF cells are distributed. This leads to an approach to calculate the actual extractable entropy using the mutual information between enrollment and reconstruction measurements. Second, a method to determine uniqueness is adopted from the field of biometrics in such a way that the creation entropy of PUFs can be estimated.

In Chapter 7, all introduced entropy estimated methods are analysed. To this end, synthetic data with known entropy is created such that the entropy estimation methods can be applied to the same dataset providing a fair comparison. Using this test, the influence of the sample size and the factors determining the entropy are investigated.

In Chapter 8, theory is put into practice by analysing PUF data from the UNIQUE project. The entropy estimation methods discussed in previous chapters will be used to analyse the entropy of various types of PUFs.

Chapter 9 concludes this thesis by summarising the main contributions, recapitulating the entropy estimation results, mentioning limitations and providing suggestions for future work.

Chapter 2

Preliminaries

2.1 Notation

Random variable Random or stochastic variables are denoted in capital letters.

Probabilities The probability of outcome x is written as p_x or $\Pr[X = x]$. The conditional probability of X given Y is denoted as $p_{x|y}$. Joint probabilities are written as p_{xy}.

Domain/space The space in which a random variable lives is denoted in a calligraphic font, thus $X \in \mathcal{X}$.

Error function The error function is denoted as $\mathrm{Erf}[x]$.

Covariance The covariance between random variables X and Y is denoted as $\mathrm{Cov}(X, Y)$.

Determinant The determinant of matrix Σ is denoted as $|\Sigma|$.

2.2 Definitions

Expected value Considering a discrete random variable X and a function $f(X)$ of X, the expected value of $f(X)$ is defined as:

$$\mathbb{E}[f(X)] = \sum_{x \in \mathcal{X}} \Pr[X = x] f(x). \tag{2.1}$$

Shannon Entropy Considering a discrete random variable X with probabilities $\Pr[X = x] = p_x$, the Shannon entropy is defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p_x \log p_x. \tag{2.2}$$

Differential entropy Considering a continuous random variable X with probability density function $p(x)$ and probabilities $\Pr(a \leq X \leq b) = \int_a^b \mathrm{d}x\, p(x)$, the differential entropy is defined as:

$$h(X) = -\int_{\mathcal{X}} \mathrm{d}x\, p(x) \log p(x). \tag{2.3}$$

Min-entropy Let p_{max} denote the most likely outcome of random variable X, then the min-entropy of X is defined as:

$$H_\infty(X) = H_{\min}(X) = -\log p_{\max}. \tag{2.4}$$

Joint entropy Considering discrete random variables X and Y, the joint entropy of X and Y is defined as:

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{xy}\log p_{xy}. \tag{2.5}$$

Conditional entropy Considering discrete random variables X and Y, the conditional entropy of X given Y is defined as:

$$H(X|Y) = \mathbb{E}_y[H(X|Y=y)] = -\sum_{y\in\mathcal{Y}} p_y \sum_{x\in\mathcal{X}} p_{x|y}\log p_{x|y}. \tag{2.6}$$

Mutual Information Considering discrete random variables X and Y, the mutual information of these two random variables is defined as:

$$I(X;Y) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{xy}\log\frac{p_{xy}}{p_x p_y}. \tag{2.7}$$

Considering continuous random variables X and Y, the mutual information of these two random variables is defined as:

$$I(X;Y) = \int_{\mathcal{X}} \mathrm{d}x \int_{\mathcal{Y}} \mathrm{d}y\, p(x,y)\log\frac{p(x,y)}{p(x)p(y)}. \tag{2.8}$$

Chapter 3

Physical Unclonable Functions

A physical unclonable function is a random function which can only be evaluated with the help of a physical system. A PUF is easy to evaluate, but hard to replicate physically due to uncontrollable manufacturing variations. Internal randomness ensures that the input, or challenge, with which the PUF is queried is unpredictably mapped to a response [7].

PUFs were introduced in 2001 by Pappu when he produced the physical equivalent of an algorithmic one-way function. A one-way function is characterised by being easy to evaluate on every input but hard to invert given an output. Algorithmic one-way functions play a major role in secure cryptographic schemes.

This first PUF, the optical PUF, was indeed the physical equivalent of an algorithmic one-way function. When stimulated by the coherent radiation of laser light, the optical PUF produces a speckle pattern (response). This response depends on the angle and place of entry (and possibly amplitude or wave-length) where the laser light enters the PUF medium [34]. This 'physical one-way function' was used to obtain an identifier that is unique, tamper-proof and unforgeable; very much the same properties found in traditional algorithmic one-way functions. However, the optical PUF is very difficult to integrate in electronics.

Since then, different types of physical systems that exhibit similar behaviour emerged. First silicon PUFs, where the response was obtained from statistical delay variations within an IC [13]. This PUF is very suitable to be used to authenticate integrated circuits, as they require no additions to the IC such as with the optical PUF. Another example is the coating PUF, which obtains a response, unique to every device, from measuring random dielectric particles present in special coatings [45]. This coating could be applied on top of an IC, which means that tampering with it would change the capacitance of the special coating thereby destroying the original unique identifier. This made these PUFs very suitable to protect chips against tampering.

In addition to the types briefly discussed here, a lot more PUF proposals have been made. With the addition of new proposals, also came new jargon. Unfortunately, the result is that current terminology around PUFs is very inconsistent. Often, the following terms are used to denote the same principle: Physical Unclon(e)able Function, Physically Unclon(e)able Function [4], Physical Random Function [12, 13, 14], Physical Unreadable Function, Physical Hash Function, Physical One-way Function [34], Physically Obscured Key (POK), Physically Obfuscated Key. Concerning terminology, this thesis refers to these systems as Physical Unclonable Functions as this is the most commonly used term.

Just as with terminology, the exact definition of what a PUF is varies in the literature. There, a given definition usually lists a number of requirements that have to be met for an object to

be called a PUF. However, actual PUF proposals often only partially match these criteria. In this thesis, we use the following definition:

Definition 3.0.1 *A Physical Unclonable Function (PUF) is the physical embodiment of a random function which is hard to clone and that exhibits challenge-response behaviour that is hard to characterise or model but which is otherwise easily and reliably evaluated.*

Note the word 'reliably', because in general, whenever a PUF is queried with the same challenges, the corresponding responses vary slightly. Thus PUFs cannot directly be used in scenarios where noise free output is required and therefore they require processing.

3.1 Properties

With the broad collection of PUFs known today, several properties have been identified [31, 50] that help find suitable applications and discern the quality of different PUF types. In this section, a list is composed discussing these properties. The first four properties are core to defining a PUF, without them one cannot speak of a PUF. The other properties fortify the strength of the PUF or broaden the area of application.

- Physically unclonable (or manufacturer resistant)

  A PUF is called physically unclonable (manufacturer resistant) when it is infeasible, even for the manufacturer, to produce two PUFs with identical challenge response behaviour.

- Easily evaluatable

  Given a PUF and a challenge, it is easy to obtain the corresponding response. In a theoretical context, easy means in polynomial time while in a practical context, easily evaluatable refers to effort or cost.

- Reproducible (or robust)

  Given a PUF and a challenge, querying the PUF multiple times with the same challenge leads to responses that differ only slightly, the error should be easily correctable. This property differentiates PUFs from true random number generators (TRNGs).

- Opaqueness

  The physical structure of the PUF should be hard to characterize. This means that an adversary should not be able to replicate a PUFs challenge-response behaviour by studying its physical properties.

- Uniqueness

  Responses of a certain PUF contain some information about the identity of that PUF, which uniquely identifies it in a given population. Uniqueness is implied by physical unclonability.

- Mathematically unclonable

  Given a PUF, it is hard to construct a function or (probabilistic) algorithm which maps the same challenges to the same responses as the PUF (up to a small error).

- Unpredictable

  Given a PUF and an arbitrarily large set of challenge response pairs (belonging to that PUF), it is hard to predict the response to an unknown random challenge. Unpredictability is implied by mathematical unclonability.

- Onewayness

  Given a response and a PUF, it is hard to find a challenge such that the PUF maps this challenge to the known response.

- Tamper evident

  Any small alteration of the physical entity embedding a PUF alters its challenge response behaviour in such a way, that with very high probability, responses to challenges are different from responses to the same challenges obtained *before* the alteration.

Furthermore, between different PUF types, the number of possible challenge-response pairs strongly differ. Most memory based PUF, such as the SRAM or D Flip-Flop PUF only have one CRP, where the response is just the start-up state of the cells.

Besides these properties, PUFs can further be divided into two classes, or settings. The first setting is that of the bare, or uncontrolled PUF, where a reader interacts directly with the PUF. However, for most applications, implementing a bare PUF is not secure. Instead, one needs to bind a PUF with a control algorithm. Without control, anybody can query the PUF with any challenge. A controlled PUF (CPUF) makes enumerating PUF responses harder and prevents man-in-the-middle (MITM) attacks [13][14].

Definition 3.1.1 *A PUF is a controlled PUF when it is inseparably bound with control logic through which the PUF needs to be accessed, such that if an attacker tries to circumvent the control logic, the PUF is destroyed. This control logic can then be used to control challenges fed to the PUF.*

In addition, some PUFs require external or extra logic to explicitly introduce randomness to function (such as the optical and coating PUF). However, other PUFs can be utilized using the already present logic on hardware which make them a lot cheaper and easier to implement.

Definition 3.1.2 *A PUF is called intrinsic when it is embedded in the hardware it protects, requiring no modification or additions to introduce randomness.*

Another addition is the introduction of (logically or physically) reconfigurable PUFs. These PUFs can change their challenge-response behaviour, effectively creating a new PUF [24, 27].

3.2 Secure error correction

Responses from PUFs are not free of noise. This means there will be some variations between responses gathered from the same PUF using the same challenge. In order to use these responses, these variations need to be corrected such that identical output is obtained. This especially holds when using PUF responses for cryptographic keys, since a single bit flip in a key causes entirely different results when the key is used (e.g., for encryption).

Another problem is non-uniformity, when PUFs generally favour certain outcomes more than others. For example, some latches start-up more often in the one-state than in the zero state.

In order to solve these problems, *fuzzy extractors* can be used to extract a noise-free uniform random output [11]. The goal of fuzzy extractors is to increase both robustness and unpredictability of PUF responses. This is respectively achieved by two main processes: information reconciliation and privacy amplification.

In these processes, we distinguish between enrollment measurements and reconstruction measurements. In an enrollment measurement, the PUF is queried and the response is used to obtain a secret and some helper data. With this helper data, a later measurement, known as a reconstruction measurement, can be used to reconstruct the secret, but only if the reconstruction measurement lies close enough to the original enrollment measurement. Usually, enrollment occurs at the manufacturer while reconstruction occurs 'in the field', where the PUF is no longer under control of the manufacturer.

Information reconciliation is the process of correcting errors, or differences, between PUF responses. Given two close responses, one wants to reconcile the differences, or bit errors, while preserving as much information (present in the response) as possible. Information reconciliation is used to ensure that the outputs of a reconstruction measurement matches a previously obtained enrollment measurement of the same device.

The second procedure, privacy amplification, involves extracting randomness from PUF responses. Often, responses are not uniformly distributed, or maybe an attacker knows something about the probability distribution such as the most likely outcome p_{max}. Using strong extractors, the responses can be made arbitrarily close to uniform ensuring that attackers know nothing that can lead to an advantage. Privacy amplification ensures that the secret S is unpredictable.

Fuzzy extractors work in two phases. The first phase consists of a generation or `gen` procedure and is applied when enrolling the PUF. The enrollment measurement X is fed to the `gen` procedure which extracts a secret S and helper data W. This helper data should not reveal any information about S and can be stored publicly.

The second phase, called replication or `rep`, is applied during reconstruction to reconstruct the secret $\hat{S}$ ($\Pr[S \neq \hat{S}]$ should be small) using as input the helper data W and a new measurement Y. This reconstruction only succeeds if the new measurement with noise lies within a certain threshold from X. Thus, fuzzy extractors feature error correction **and** guarantee secrecy of the key that is extracted from a PUF response. This process is displayed graphically in Figure 3.1.

Note that these processes reduce entropy; the extractable entropy is less than present in the raw PUF responses. It would be too involved to discuss every error correction method in detail to give an estimation of the resulting entropy. Therefore, the entropy analysis in this paper only considers raw responses, thus before any processing steps such as fuzzy extractors. This keeps the thesis general and provides a reliable estimate not depending on specific implementations.

3.3 PUF Applications

The optical PUF by Pappu was used to obtain a unique identifier. However, PUFs can be used for a wide range of applications. In this section the most important PUF applications are shortly discussed.

Anti-counterfeiting Traditionally (consider paper money for example), an authenticity mark is added to the product which is hard to forge. The problem with this approach is that all marks are identical, meaning that once an attacker succeeds in cloning one mark, the product can effectively be forged.

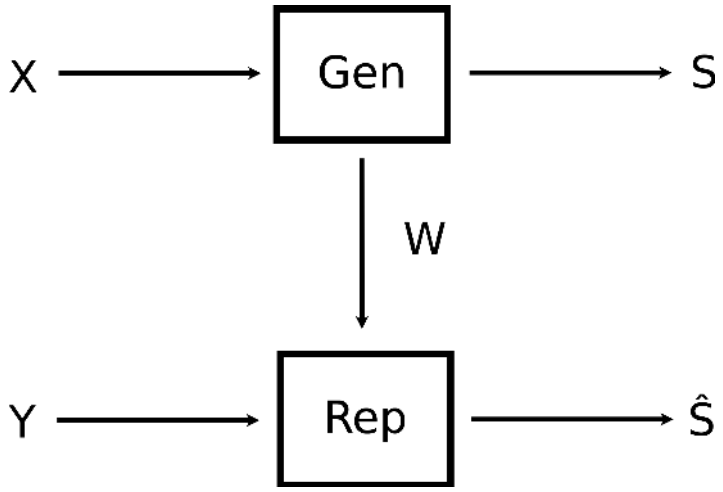


Figure 3.1: Illustration of a fuzzy extractor. The generation procedure constructs helper data W and secret S from enrollment measurement X. The reconstruction procedure reconstructs the secret $\hat{S}$ from reconstruction measurement Y using the helper data.
For a reconstruction measurement Y similar to enrollment measurement X, the probability of reconstructing a secret $\hat{S}$ that is not exactly equal to S is negligible.

PUFs can be added to a valuable object as an unique authentication token, ensuring the genuineness of an object. Since every PUF is unique and PUFs are physically unclonable, it is impossible to create a counterfeit copy passing as a genuine one [45]. For this application, implementing a bare PUF is enough and only the core properties, physical unclonability, robustness and easily evaluatable are of importance.

IP Protection For integrated circuits, it is very easy to clone (and modify) existing products by obtaining the design plans (intellectual property) either through reverse engineering, industrial espionage or unreliable outsourcing companies [16]. Once the design plans are no longer secret, illicit products can be produced and sold below the marked price since the original developer paid for research and development cost.

Reconfigurable integrated circuits, such as field-programmable gate arrays (FPGAs), are especially vulnerable to this kind of counterfeiting. FPGAs are configured after manufacturing using a so called bit-file, which contains the intellectual property (IP) of the designer. This bit-file is often stored in non-volatile memory, which is easy to tap allowing attackers to steal the design plans. They can then configure their own FPGAs to obtain counterfeit copies of the product [17].

Using PUFs, it is possible to defend against this kind of attack by encrypting the bit-file with an FPGA specific key. This key can be derived from an intrinsic PUF, for example a DFF [30] or SRAM PUF [17].

Device authentication The validating party securely stores for each PUF a collection of challenge-response pairs. When authentication is required, a challenge is fed to the device that needs to be authenticated. If the response of the device matches the response stored in the database for that particular challenge, the device is authenticated.

An example usage is a smart card based authentication system. Since the PUF within the card is physically unclonable, even an attacker who possesses the card for an arbitrary time period

cannot clone it to spoof authentication. This means a lost and found card can be reused, whereas normally a new card would have to be issued [29].

Secure key storage In order to provide services such as Pay TV, DRM or even file encryption, hardware devices need to store a device unique secret key. Often these keys are stored in physical memory during or after the manufacturing process. For example, smart cards used for these purposes nowadays contain a digital key in EEPROM (or other memory) which is prone to various attacks such as simple and differential power analysis, power or clock glitching and software exploits which dump memory [5]. Using a PUF, storing a digital key is no longer necessary since the PUF itself functions as the secret key. PUFs used in this scenario are often denoted as physically obscured keys.

Certified Execution When a controlled PUF is coupled to a computation device, secure execution is possible. This requires that the verifier knows a set of CRPs. Besides the code, the device is offered a PUF challenge. The device performs the computation and returns the result as well as a message authentication code (MAC) over the result computation and the PUF response [13].

In this way, the verifier can check the authenticity of the computation by calculating the MAC over the stored response and the received computation result and comparing this to the obtained MAC.

Note that to do this securely, the PUF needs to be a controlled PUF otherwise an attacker can eavesdrop the PUF response and use this to calculate a MAC over his own forged result which the verifier would then accept as genuine [13].

3.4 Attacks on PUFs

In this section, possible attacks against PUFs are briefly discussed.

3.4.1 Difficulty of random duplication

To create a counterfeit product, an attacker can attempt to fabricate a clone containing a PUF with the exact same type as in the original product. Assume that the attacker also has the design plans of the product, including masks of the IC and the specification of the PUF. The statistical variation in PUF fabrication ensures that for an attacker to successfully create an identical PUF, depending on the PUF's entropy, a large number of ICs need to be fabricated in order to discover a suitable counterfeit.

Let us denote the entropy in bits b. This means there are potentially 2^b number of different PUFs possible. Here we assume that all the possibilities occur with equal probability. We also introduce k as the number of valid genuine PUFs the attacker knows, so if a fabricated clone matches one of these k PUFs, the attacker can successfully create a counterfeit product with this clone.

If we define a as the number of PUFs the attacker produces (at random), it is possible to bound the number of successfully created cloned PUFs as (based on the birthday collision attack):

$$s = \frac{a \cdot k}{2^b}. \tag{3.1}$$

If the cost of creating a PUF is expressed as c and the profit of a successful counterfeit product as p, it is straightforward to calculate when cloning is profitable:

$$p \cdot s > a \cdot c =$$
$$\frac{p \cdot k}{2^b} > c.$$

Which means the required entropy to make the attack unprofitable is:

$$b > \log k \frac{p}{c}. \tag{3.2}$$

This examples illustrates the importance of entropy in PUF responses; the larger this entropy the higher the cost for an adversary to find a successful clone[50].

3.4.2 Modelling attacks

Modelling attacks try to indistinguishably mimic the challenge-response behaviour of a PUF, e.g. by training a computer algorithm on a polynomial number of challenges. Attacks based on additive delay models [15] and machine learning through logistic regression [35] have already been suggested for arbiter and ring oscillator PUFs.

PUFs with multiple CRP should be resistant against modelling attacks, which means that if an attacker obtains an extensive set of CRPs, the response to an unknown challenge should still be unpredictable. However, it has already been proven that this is not the case for arbiter PUFs. Hospodar et al. proved that obtaining 500 CRPs was enough to create a 90% accurate model [21]. Note that this may fall well within error correction bounds, which means the model is accurate enough to pass as the actual PUF in practice. To model the PUF perfectly without relying on error correction, a factor 10 more CRPs is enough. This indicates that indeed the arbiter PUF is very susceptible to modelling attacks.

3.4.3 Side-channel attacks

In this attack scenario, the attacker tries to obtain knowledge about the PUF response by measuring additional effects such as electromagnetic radiation (EM), power consumption or timing information that is leaked when operating the PUF.

However, the attacker cannot directly measure (for example, delays) inside the PUFs, as these measurement requires the attacker to open the packaging and remove some layers, therefore becoming an invasive attack (which are handled in the next section).

There is research investigating the susceptibility of PUFs to side channel attacks, though all these attack seem to target fuzzy extractors and not the actual PUFs itself [38, 23]. By examining the power leakage occurring in the error correction phase, it is possible to recover PUF responses [23]. However, we are not aware of any practical side-channel attacks on actual PUF implementations.

3.4.4 Invasive methods

Invasive attacks include the removal of chip layers to perform direct measurements, shorting circuits using focused ion beams or probing underlying wires. Currently, these attacks still influence

the behaviour of the PUF, thereby destroying the unique PUF fingerprint. This means that even when the attacker is able to measure delays or the contents of memory cells, the intrinsic variations have been altered changing the PUF behaviour (identity) [46]. Hence, these attacks are *currently* not sophisticated enough to directly read out PUF responses. However, semi-invasive attacks based on EM cartography have been proposed for the ring oscillator PUF [33].

3.5 Reliability

For a PUF to be reliable, or robust, it means that, even when challenging the PUF under varying (environmental) conditions, the reconstruction measurements should not differ significantly from the earlier enrollment measurement. If they do, fuzzy extractors are no longer able to reconstruct the correct secret.

The most influential environmental factors effecting reliability are shortly explained here, as well as how reliability can be determined.

3.5.1 Temperature

The environmental temperature has an effect on on-chip junction temperatures which in turn effects the delay of gates and wires, thus possibly changing PUF behaviour [29].

Typically, we see that with D flip-flops for example, the number of ones in PUF responses decreases with a rise in temperature, while SRAM memories are more robust to changes in temperature [8].

3.5.2 Voltage

Another aspect that has a major impact on the behaviour of certain PUFs is the supply voltage. Here it is possible to distinguish at least two different effects, the actual supply voltage and the time it takes to establish this voltage which is known as the voltage ramp-up time.

Decreasing the supply voltage will slow down the circuit. This affects the reliability as different parts of the circuit suffer from a non-linear performance loss [8]. Also, increasing the voltage ramp-up time of both DFF [8] and SRAM [8, 39] PUFs seems to reduce reliability. The underlying reason(s) for the performance change under increasing ramp-up times are currently not well understood.

3.5.3 Ageing

The main environmental factors causing changes in PUF response behaviour are operational temperature and voltage. However, electrical components also degrade over time. Several causes have been identified such as electro migration (EM), hot carrier injection (HCI), time-dependent dielectric breakdown (TDDB) and most prominently negative bias temperature instability (NBTI) [19][37]. Therefore, ageing tests are performed to see how PUF responses change over prolonged use. These tests work by increasing the temperature and supply voltage to accelerate the ageing effect of the chip, and then taking measurements (at reference environment circumstances) over a prolonged period of time. The total acceleration factor (TAF) of this process can be computed by:

$$TAF = \exp(E_g/k(1/T_{nop} - 1/T_{stress}) + \gamma(V_{stress} - V_{nop})). \tag{3.3}$$

In this formula, E_g is the activation energy, k is the Boltzmann constant and γ is the voltage exponent factor. T and V denote respectively the temperature and the voltage where the subscripts 'nop' and 'stress' label the normal operation conditions and the conditions under stress. Typically, an ageing acceleration factor of well over 10 times can be achieved [8, 39, 41].

3.5.4 Reliability assessment

The most used and straight-forward method to determine reliability is by examining Hamming distances between measurements of the same device subject to environmental variations. This provides an estimate of how much variation there is within PUF responses over time, hence it is also known as intra-device distance.

Temperature effects are investigated by placing a PUF in a climate chamber which is usually set to vary between the industrial standard operational limits, which is between -40 and $+85\,^{\circ}\mathrm{C}$.

Robustness to changes in supply are typically measured by varying the supply voltage levels between $90\% - 110\%$ of Vdd. Voltage ramp-up times are simply tested by varying (mostly, increasing) the time it takes to establish the supply voltage.

3.6 Security

The main measure for determining the security or uniqueness of a PUF, is entropy. In the PUF context, it is a measure of the missing information, or the uncertainty, about an unknown response. Entropy is a very important concept when dealing with PUFs, as it bounds both the maximal length of a cryptographic keys that can be derived as well as the maximum number of uniquely distinguishable PUFs.

In the information-theoretic definition, entropy is a measure of uncertainty associated with a random variable. To calculate the entropy of a PUF, the PUF itself can be abstracted away as a random variable, which is explained in more detail in Chapter 5.

3.7 PUF types present on the UNIQUE ASIC

Currently, there is broad scala of PUF types available [31]. The most interesting ones, from a practical and commercial perspective, have been brought together by the UNIQUE project to objectively evaluate their performance [47].

The result is an ASIC containing six different types of PUFs, four of which are memory based and two based on delay times. In addition, the ASIC features an active core which generates on-chip switching behaviour simulating a more realistic operating environment.

In this section, an introduction is given to each PUF type, their source of randomness is discussed and the PUF properties are evaluated.

3.7.1 Memory Based PUFs

Memory based PUFs are usually based on the principle that memory cells can be in a logically unknown state, which means that the stable state a memory cell converges to is not known beforehand. This is achieved by cross-coupling two inverter elements.

The value a memory cell converges to is dependent on the (variation between) threshold voltages of the inverters. This variation in threshold voltage is caused by intrinsic differences in

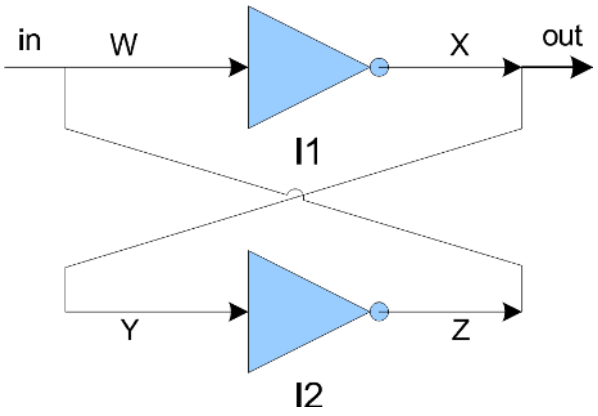


Figure 3.2: Figure displaying cross-coupled inverters, the essential element of memory based PUFs. Image taken from Kumar et al. 2008 [26].

the electronic elements. As this is beyond the control of the manufacturer, physical unclonability is achieved. A schematic illustrating this principle is displayed in Figure 3.3.

An advantage of memory based PUFs is that they directly produce a binary string from measuring the start-up values of the memory, there is no quantization step needed [18]. On the other hand, this also means that there is only 1 challenge-response pair (a term that is hardly applicable), which is reading out this binary string.

SRAM PUF

SRAM, Static Random Access Memory, is a type of volatile memory that was first considered as a PUF by Guajardo et al. [17] in 2007. An SRAM cell consists of six transistors, two transistors are used for access (read/write operations) and the other four form the storage cell. This storage cell, which is formed by two cross-coupled inverters, has two stable states allowing storage of a single bit. However, when a cell powers up, the initial state is undefined but a cell always has a certain preference to start up in either the zero or one state [18].

This preference is the result of a mismatch of the threshold voltage, the voltage level on which a transistor switches. This mismatch originates from the creation process and cannot be controlled by the manufacturer. Because different cells have different thresholds and therefore different start-up preferences, start-up behaviour becomes device specific.

The four transistors keeping the state of the memory cell are constructed weak in order to easily set the stored value to zero or one. A consequence is that the memory cell is extremely vulnerable to the voltage threshold mismatch caused by microscopic variations in the dopant atoms in the channel regions of the transistors [2][6].

During start-up, an SRAM cell has no input from externally exerted signals, which means the sole reason for converging to a certain start-up state is caused by intrinsic variations of the transistors. Therefore, the start-up state preference can differ among cells but is generally stable across multiple power-ups [16].

An SRAM PUF is built from standard technology components with all cells clustered near each other. The advantage is that this is cheap and takes very little space. This however also has a drawback namely that these memory cells are very easily located on a chip. This reduces the effort an attacker has to spend looking for the PUF on a chip in order to tamper with it. An illustration of this is shown in Figure 3.5.

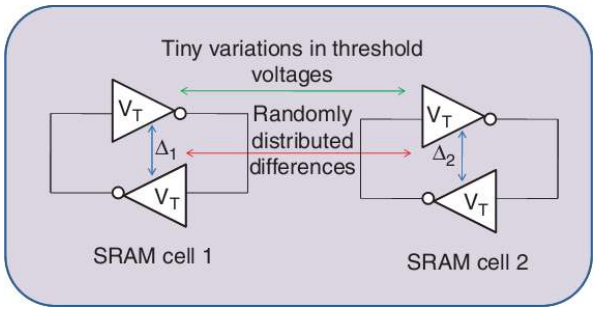


Figure 3.3: An illustration of two SRAM cells displaying their threshold mismatch Δ_1 and Δ_2. This threshold mismatch determines the stable state the cell converges to. Figure from Handschuh et al. [19].

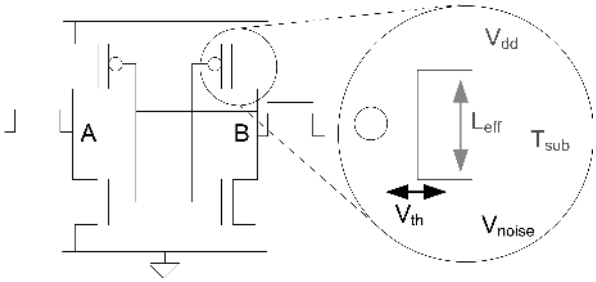


Figure 3.4: Typical SRAM cell indicating the threshold mismatch on the transistor level. Picture by Holcomb et al. [20].

On the UNIQUE chip, four instances of 8KB TSMC SRAM memories are present. The total area occupied by these memories is 0.213 mm^2 [25].

Latch PUF

A **latch** is an electronic circuit that can hold two values, effectively storing one bit. It holds the logic value that was last applied to its input. It is mostly used as a transparent storage element, which means that a change of input immediately effects its output. A flip-flop is the non-transparent counterpart of the latch[1] , in which changes propagate on clock signals only.

The **latch PUF** was discovered by Su and Holleman in 2008 [42]. A latch PUF consists of two cross-coupled NOR-gates (or NAND-gates) and is based on a similar principle as the earlier introduced SRAM storage cells, in that they can hold two values but that before a stable state is set, they are in a logically unknown state. With Latch PUFs, this behaviour can be triggered by asserting a reset signal, eliminating the need to restart the chip. However this feature is optional, on the UNIQUE chip this is not implemented.

The internal mismatch of the electronic components cause the latch to converge to a stable state, which is very similar to what was seen by SRAM PUFs (and all memory based PUFs).

The UNIQUE chip has four 1KB latch PUF instances, two of which are addressed using MUX-based addressing and two using a scan-chain method. The latter method is an attempt to reduce addressing logic, but unfortunately there were problems with the read-out circuitry of these instances [25]. Therefore, in the entropy analysis of the latch PUF, only the two instances with

[1] In this thesis this distinction will be kept, but the terms latch and flip-flop are often used interchangeably.

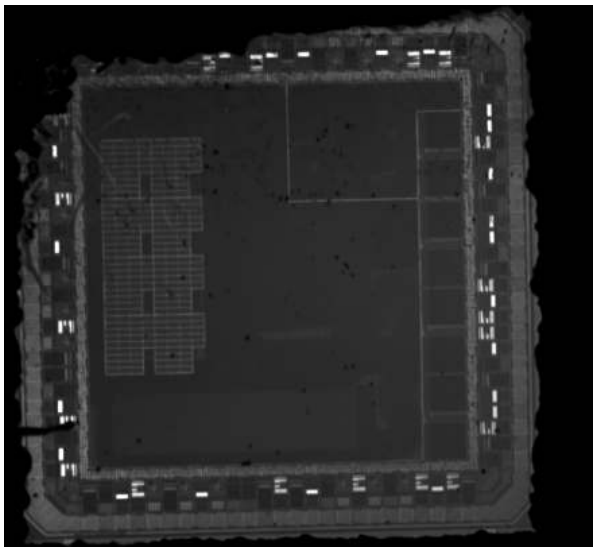


Figure 3.5: A back side IR image of the UNIQUE chip where the packaging and top layers been removed. On the right side, the SRAMs are clearly visible as well as the arbiters left and the ring oscillators at the bottom. Picture taken by and property of Thales Communications & Security.

MUX-based addressing are considered.

The four latch instances occupy a total area of 0.272mm^2 and provide 8192 bits of output each [25]. An advantages of latches is that they need not be clustered together on a chip. On the UNIQUE ASIC, some latches are placed clustered while others are distributed over the chip.

D Flip-Flop PUF

The DFF PUF was introduced by Maes et al. in 2008 as a replacement for the SRAM PUF on FPGA boards [30]. It functions exactly the same as an SRAM PUF except the memories are using D flip-flops instead of SRAM cells. D flip-flops, like latches, can be placed freely over the chip.

On the UNIQUE ASIC, the DFF PUF is based on DFF memory cells realised by two latches. The ASIC features four of these DFF memories that are 1KB in size, thus also yielding 8192 bits of output each. The area occupied by the four DFF instances is 0.392mm^2 [25].

Buskeeper PUF

The buskeeper PUF is the most recently introduced PUF type implemented on the UNIQUE chip, invented by Simons et al. in 2012 [41]. A buskeeper, or busholder, is a component used to prevent on-chip buses from floating. Floating buses increase the power consumption of a chip. A buskeeper prevents this by maintaining the last driven state of the bus. A buskeeper has a low drive strength and is thus easy to override whenever data has to be send over the bus [41].

Functionally, a buskeeper is equivalent to a D latch with the enable signal connected to the power (Vdd). Structurally, a buskeeper is just a set of cross coupled inverters implementing a weak latch. A buskeeper contains no write logic and thus no control signals, resulting in a very small cell structure. Another advantage is that buskeepers can freely be distributed over the chip in contrast to SRAM memory. A high-level and transistor level overview of the buskeeper cell structure can be seen in Figure 3.6.

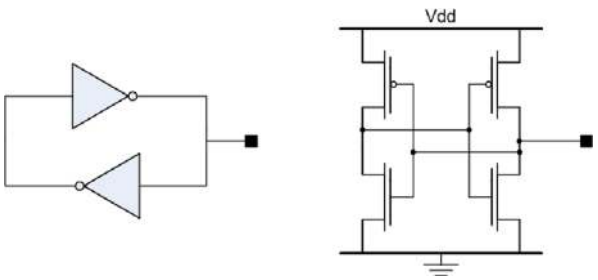


Figure 3.6: Left: high level view on buskeeper cell structure. Right: transistor level view of buskeeper. Image from Simons et al. [41].

The buskeeper PUF, as with all memory based PUFs, has two stable states. Uninitialized, the settling state of the cross-coupled inverters can be read out using a MUX tree. Observing the settling state of a batch of buskeepers leads to the PUF response.

The UNIQUE chip has two 1KB buskeeper arrays. The total area occupied by these buskeepers is 0.076mm^2 [25].

3.7.2 Delay Based PUFs

Even though it is possible to mass produce integrated circuits which function exactly the same, every single IC has unique delay characteristics as a result of the manufacturing process [28]. These delay characteristics can be used to obtain a random output suitable as a PUF response. However, the delay of a wire or a transistor is also dependent on its environment, which consists of the operation voltage, temperature but also ambient noise and ageing. Therefore, to be secure, the random delay variations need to be large enough to distinguish ICs fabricated on the same wafer, making the PUF *manufacturer resistant*. Otherwise, attackers can predict the behaviour of a PUF.

The UNIQUE ASIC features two PUFs based on timing differences. One of these PUFS, the arbiter PUF, supports multiple CRPs in contrast to all the other PUFs present on the ASIC.

Arbiter

The idea of extracting randomness from delay behaviour was utilized by Lim et al. [29] who introduced the arbiter PUF in 2004. The arbiter PUF consists of a number of special circuits that race a (simultaneously fed) rising edge along two delay paths. Each circuit has an arbiter that determines which path is fastest, returning either a zero or one.

An arbiter PUF consists of two main elements: an arbiter component with two inputs and a switching circuit. The arbiter component determines what input is high first and depending thereon outputs either a zero or a one. The switching circuit consists of a multitude of in series connected switch delay elements and forms the basis of the arbiter PUF [29].

These switch components each have two inputs and two outputs. Which input is connected to what output depends on a single challenge bit. If the challenge bit is 0, the inputs are connected straight through while if the challenge bit is 1, the paths are crossed which can be seen in Figure 3.7.

To use this arbiter circuit, a rising edge is fed to the two inputs simultaneously. Based on the challenge, which is a set of bits determining which switch circuit should switch, the path the edge

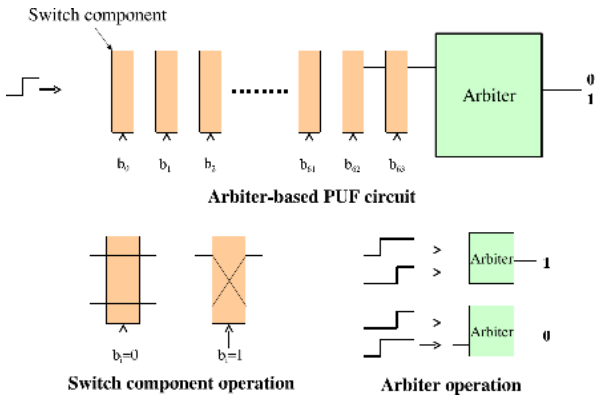


Figure 3.7: Arbiter PUF with switching components and arbiters shown. Image from Lee et al. [28].

follows changes. Depending on which input of the arbiter component the rising edge arrives at first, a zero or one is returned.

An arbiter PUF consists of multiple arbiter circuits which are fed with the same challenge. If these circuits are symmetric regarding their delay paths regardless of the challenge, and if the arbiter is unbiased, the result is unknown depending only on during manufacturing introduced variation in delay times. Thus, to maximise entropy it is vital to produce the paths as equal as possible.

An arbiter PUF with n challenges, has 2^n possible delay paths. However, these are not independent. Worse, it is shown that once an attacker knows a certain amount of CRPs, the PUF becomes completely predictable using model building attacks (see Section 3.4.2).

On the UNIQUE chip, an arbiter PUF is implemented using 256 arbiters with 64 delay elements, each corresponding to one challenge bit. Because of the importance of symmetric paths, the PUF layout is custom designed. The space occupied by the arbiter PUF is 0.213mm^2 [25].

Ring oscillator

The last PUF present on the UNIQUE chip is the ring oscillator PUF. This PUF was first implemented by Suh and Devadas [43] based on the ideas Gassend [13] posed in 2002. Compared to the arbiter delay PUF, a ring oscillator PUF is more reliable and is easier implemented [43, 7]. However, the ring oscillator PUF is also slower, uses more power and occupies more space on a chip.

The ring oscillator PUF consists of a number of identical self-oscillating circuits where the period of the oscillation is determined by uncontrollable manufacturer variations. Given two of these circuits, a zero or one is produced by comparing which circuit oscillates faster.

Given that the PUF has n of these ring oscillators, there are $n(n-1)/2$ distinct pairs to compare. However, these pairs are not independent. If one knows that oscillator A is faster than B, B is faster than C, then A will also be faster than C. To avoid correlation, usually only pairs of oscillators are compared, resulting in $n/2$ comparisons, and thus (independent) bits of output [43].

To be more robust, one should compare pairs of oscillators whose frequencies lie far apart. This ensures that environmental variations, which can impact one oscillator more than another due to different device or physical parameters, have less effect on the oscillator frequencies decreasing the probability of bit flips and thus increasing PUF reliability.

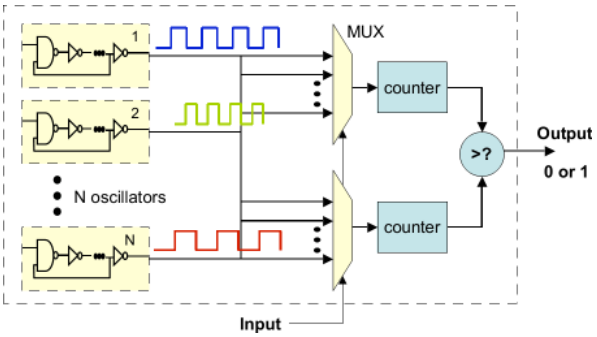


Figure 3.8: A ring oscillator PUF obtains a bit as output by comparing oscillation times. Image by Lee et al. [28].

The advantage of ring oscillator PUFs in contrast with the arbiters is that they are very easy to replicate (as a hard macro) during chip design while there is also no need for careful layout routing. However, ring oscillators are also not standard components and thus still require a semi-custom design.

On the UNIQUE chip, a ring oscillator PUF is implemented using 4096 testable oscillators arranged in 256 rows and 16 columns. Sixteen parallel measurements can be done, where only neighbouring oscillators are compared yielding 15 bits per column for a total of $256 \cdot 15 = 3840$ response bits. The space occupied by the ring oscillator PUF is 0.241mm^2 [25].

3.7.3 Overview

All 6 PUF types adhere to the four core properties defining a PUF. In general, the PUF types on the ASIC differ only slightly in PUF properties. All PUFs present satisfy uniqueness, unpredictability and are intrinsic. On the other hand, all PUFs do not satisfy mathematical unclonability and all except the ring oscillator are not one-way. An overview can be found in Table 3.1.

PUF Type	Uniqueness	Mathematically Unclonable	Unpredictable	Onewayness	Tamper evident	Nr. of CRPs	Intrinsic
SRAM	Yes	No	Yes	No	?	1	Yes
Latch	Yes	No	Yes	No	?	1	Yes
D Flip-flop	Yes	No	Yes	No	?	1	Yes
Buskeeper	Yes	No	Yes	No	?	1	Yes
Arbiter	Yes	No	No	No	?	2^{64}	Yes
Ring oscillator	Yes	No	No	?	?	1	Yes

Table 3.1: Overview matrix of various PUF types against PUF properties. The data is taken from Maes et al. [31].

It is very valuable to know how many bits a specific type of PUF can derive from a certain surface area. In the following table we provide an overview of how much area on the UNIQUE

chip surface the different PUF types occupy and how many bits can be generated. Note that these bits might be correlated, biased or otherwise not have full entropy. Later in this thesis we reconsider this table by calculating the amount of extractable entropy (in bits) per surface area.

PUF Type	Area (mm^2)	Composition	Bits per mm^2
SRAM	.213	$4 \cdot 65536$ SRAM Cells	≈ 1230723
Latch	.272	$4 \cdot 8192$ Latches	≈ 120470
D Flip-Flop	.392	$4 \cdot 8192$ D Flip-Flops	≈ 83592
Buskeeper	.076	$2 \cdot 8192$ Buskeepers	≈ 215579
Arbiter	.279	256 64-bit Arbiters	N/A
Ring oscillator	.241	4096 Inverter Chains	≈ 15934

Table 3.2: Overview of the area occupied on the UNIQUE chip surface, broken down to PUF type. Area and composition data copied from Koeberl et al. 2012 [25].
The reason for the 'not applicable' by the arbiter PUF stems from the fact this PUF has an exponential number of challenges and therefore a very large number of output bits (although these are not independent and the actual amount of bits that can be extracted is very low, consider Section 3.4.2), which cannot fairly be compared to the other PUFs.

Chapter 4

Previous work on entropy estimation

In this chapter, previous work regarding entropy estimation is considered. We found that literature containing thorough entropy analysis is scarce. Most PUF proposals only briefly mention security properties, and an actual entropy estimation is rarely seen. There is an extensive entropy analysis of the optical PUF by Tuyls et al. [44, 22] and another of the coating PUF by Škorić et al. [51], but the methods presented there are not applicable to memory-based PUFS, where the main focus of this thesis lies.

However, there are some papers providing basic uniqueness estimates. In this chapter, these mostly straightforward methods to estimate uniqueness are discussed. In addition, several methods found in the literature that actually estimate entropy and are applicable to memory based PUFs are handled as well.

4.1 Simple (Hamming) weight and distance measures

Common methods to quickly and easily determine uniqueness involve the calculation of Hamming weights and distances. The Hamming weight of a PUF, the number of cells that return non-zero upon start-up, can be used to determine if a PUF is biased [36, 48]. When sampling multiple PUFs, the minimum or maximum Hamming weight can be used to estimate an upper bound on the bias.

Often, the fractional Hamming weight is used where the weight is divided by the number of cells. It is desirable for PUFs to have a fractional Hamming weight of 0.5 since this indicates that the PUF is unbiased; on average its cells have no preference for a certain start-up state. If this is not the case, the resemblance between different PUFs will increase which negatively impacts uniqueness.

Hamming weight can also be calculated for cells, where the Hamming weight for a certain cell is summed over multiple PUFs. This can be used to determine if specific cells favour a power-up state, identify groups of correlated cells or analyse the effects of a cells physical location on the chip on its bias.

Intra-device distance The intra-device distance, also denoted as within-class or environmental variation, expresses how much a response of one device varies in multiple start-ups under possibly varying environmental conditions [43]. It provides an indication of resilience against environmental fluctuations and of general reliability (and not directly of uniqueness); intra-device distance is a measure of the average noise on responses [8, 30].

The intra-device distance is determined by calculating the (fractional) Hamming distance between enrollment and reconstruction measurements performed on the same device. The Hamming distance between two responses, interpreted as bit strings, is a count of the number of disagreeing bits.

The Hamming distance is easily calculated by applying an `xor` operation on two bit strings. This operation detects the bits that disagree (for a certain cell) which are sometimes denoted as bit errors. Simply summing these bit errors lead to the Hamming distance. Often this Hamming distance is divided by the length of the bit strings, obtaining the fractional Hamming distance. It is desirable to have a fractional Hamming distance close to 0 which means that a PUF is very reliable: even when environmental conditions change, the responses stay (almost) the same.

Inter-device distance The inter-device (or between-class) distance is a measure of the uniqueness between PUFs, it indicates how easy it is to distinguish or identify different devices [43]. This measure can again be calculated with (fractional) Hamming distances, although now between different devices by considering all pairwise comparisons.

For uniqueness, it is desirable to have an inter-device distance close to 0.5 which means that on average half the cells prefer a different start-up state. It indicates a low correlation [39, 36] between responses of different devices and as such gives the least guarantees to predict the behaviour of a certain PUF given a collection of other PUFs.

4.2 Daugman method

The field of biometrics concerns the study of recognition and identification of individuals based on measurable physiological or behavioural characteristics. Examples are gait analysis and fingerprint, iris or face recognition. The study of PUFs and biometrics exhibit resemblances especially in recognition based on the discrimination of certain features.

Here, a method is described from the field of biometrics which is viable to be used in PUF uniqueness analysis. In 2001, John Daugman developed a method to perform identification based on iris recognition [10]. Using Gabor kernels, phase information is extracted from the iris. This transform yields 2048 phase bits per iris along with an equally long (bit)mask signifying areas that are unusable due to illumination, contrast or other issues [10].

The principle on which the iris recognition algorithm operates is that recognition of an iris is a failure of a test of statistical independence. Given enough degrees of freedom, this test is guaranteed to pass for different eyes but fails when comparing phase information belonging to the same eye.

For the implementation, consider comparing two eyes given their phase information and bit masks denoting unsuitable bits. The masks are applied to the phase information to filter out the 'corrupted' bits. The resulting bit strings are compared using the Boolean exclusive OR operation, `xor`, to detect the number of disagreeing bits. This number is divided by the length of the bit mask, obtaining a ratio signifying the degree of dissimilarity or distance of the two strings, almost similar to the inter-device distance discussed above. When this ratio is (very) low, the test of statistical independence is failed which leads to the conclusion that the phase information belongs to the same eye. For two different eyes, one expects a dissimilarity ratio (fractional Hamming distance) close to 0.5.

Daugman collected data to perform over 9 million comparisons and concluded that the Hamming distances are binomially distributed with an observed mean of 0.499, extremely close to the

expected 0.5. This conclusions stems from the fact that comparing phase bits corresponds to a Bernoulli trial, where consecutive trials may be correlated.

Correlated Bernoulli trials lead to a reduction in the (effective) number of trials but these are still binomially distributed. This effective number of trials, the degrees of freedom, can in a way be seen as an uniqueness measure.

In Section 6.3, a method is proposed to apply the Daugman routine to PUF data. Here, PUF responses take on the role of the phase data. Optionally, the bit masks as discussed above can be used as creating helper data indicating the unstable cells in the response.

4.3 Min-entropy estimation using fractional Hamming weight

The min-entropy is defined as the log of the most likely outcome, which in the context of PUFs corresponds to guessing a response correctly in one go. This is a very conservative entropy estimation, but a good one for measuring uncertainty about a cryptographic key.

In the literature, the min-entropy is used to establish a lower bound on the creation entropy [41]. To determine the min-entropy, for each cell the fractional Hamming weight (over all sampled PUFs) is determined. This provides an estimate of the probability of a cell to start-up in the one (p_1) or zero (p_0) state. The maximum probability of these two is used in the maximum entropy formula (Equation 2.4) to calculate the min-entropy of the cell. This procedure is repeated for all cells, and after averaging, gives the total min-entropy per cell. This method should only be used if the cells are independent, which usually is assumed [41, 8].

4.4 Compression using the context-tree weighting algorithm

Data compression is strongly related to entropy; entropy provides a lower bound on the average length of the shortest description of the data [9]. Relating this to PUFs, it means that an optimal compression algorithm can compress a PUF response to a description with length at least equal to the entropy of the PUF data. By reversing this principle, an optimal compression algorithm can be used to provide an estimate for the PUF entropy.

In PUF entropy analysis, the 'context-tree weighting algorithm' (CTW [52]) is commonly used to estimate entropy [22, 4, 8, 36]. CTW is an optimal compression algorithm for stationary ergodic sources. With PUFs, this means that the statistical properties of the responses do not change over time and that these properties can be deduced from a single, sufficiently large, sample.

4.5 Discussion of methods

The discussed Hamming weight and distance measures provide an indication for uniqueness, but do not provide a concrete entropy estimation. Although these methods can be used to identify effects that reduce entropy, e.g. Hamming weight for biasing and Hamming distance to compare PUF similarity, they do not provide a quantification of the entropy loss of these effects.

The Daugman approach, min-entropy estimation method and the CTW compression test can be used to calculate an actual entropy estimation. However, with the Daugman approach it is not entirely clear how it statistically works, and moreover, the distances compared are not independent. Furthermore, min-entropy is always less than or equal to the Shannon entropy

with equality only in the case that the distribution considered is uniform. As such, the min-entropy method provides a lower bound on the creation entropy, although it is not clear what the magnitude of this under-estimation is. A similar estimation accuracy issue arises with the CTW compression test, which also estimates the creation entropy but where the overhead of the compression model is unclear.

Furthermore, all three methods provide an entropy estimation only in case of a single enrollment measurement, making the methods unusable if multiple enrollment measurements are used. However, the min-entropy and the Daugman method can easily be extended to calculate entropy using multiple measurements.

Lastly, note that all methods discussed here determine the creation entropy, i.e. the entropy present in enrollment responses, which means they omit the conditions (noise) under which the PUF is used in practice. Therefore, these entropy estimation results might not actually be an accurate indication of the uncertainty in PUF responses when they are used in the field. In the following chapters, we will extend entropy estimation with the inclusion of the conditions under which the PUFs are challenged. In addition, the number of measurements used during enrollment and reconstruction will be included to keep the analysis as generic as possible.

4.6 Summary of entropy estimation results found in the literature

To conclude this chapter, previous results from uniqueness and entropy estimations of the PUF types present on the UNIQUE ASIC are presented and discussed.

All PUF proposals and evaluation papers determine the intra and inter-device distance under enrollment conditions, which means that all responses are gathered under room temperature and at normal operating voltage. Most works also considered the intra-device distance under varying environmental conditions to determine the robustness. This usually consists of changing the operation temperature to the limit of industrial requirements (-40 and $+85\,^{\circ}$C) and changing the supply voltage with $\pm 10\%$.

Notably, most PUF evaluations did not compare inter-device distance under varying environmental conditions. Instead, reliability and uniqueness are often assessed separately. Still, an inter-device distance test between enrollment and reconstruction measurements may be useful to investigate the influence of the reconstruction environment on the uniqueness, especially since this environment is usually not under control when PUFs are used in practice. For example, it may be interesting to learn if PUFs responses become more alike when temperature rises, reducing the length of an identifier that can be extracted from the PUF.

More recent works also calculated the CTW compression ratio and the min-entropy per bit, providing a measure for entropy in PUF responses. Again, these measures are usually only calculated under enrollment conditions.

The results are presented in Table 4.1. The left-most column displays the PUF type evaluated and provides a reference to the related work. Most evaluations come from either PUF proposal papers or evaluation projects such as UNIQUE (which is also indicated in the first column). The other columns denote the test that was performed.

Type	Intra-device Hamming distance					Inter	Min-entropy	CTW Ratio
	E	$T-$	$T+$	$V-$	$V+$			
Latch								
UNIQUE [25]	<0.04	<0.28	<0.18			0.37		
SRAM								
Proposal [17]	<0.04	<0.12	<0.12			0.5		76
UNIQUE [25]	<0.06	<0.08	<0.08			0.50		
NXP [8]	0.06	<0.1	<0.09	<0.07	<0.07	0.49	0.75	99.1
TSMC [8]	0.05					0.50	0.76	100
90nm [39]	0.05	<0.19	≈0.08	≈0.06	≈0.06	≈0.50	0.76	100
7 SRAMs [36]	0.02-0.06			0.02-0.06	0.02-0.06	0.45-0.50		98.2-100
DFF								
Proposal [30]	<0.01					>0.1		
UNIQUE [25]	<0.04	<0.17	<0.21			0.42		
130nm [48]	<0.07	<0.13	<0.13			0.36		81.3
65nm[8]	0.04	<0.2	<0.4	<0.07	<0.07	0.50	0.77	100
Buskeeper								
Proposal [41]	≈0.04	≈0.1	<0.2	<0.05	<0.05	0.50	0.82	99
Arbiter								
Proposal [29]	0.01		0.05		0.04	0.23		
UNIQUE [25]	<0.04	<0.5	<0.5			0.47		
RO								
Proposal [43]	-	<0.01	<0.01	<0.01	<0.01	0.46		
UNIQUE [25]	<0.03	<0.04	<0.04			0.50		

Table 4.1: This table provides an overview of the results of uniqueness and entropy analyses found in the literature. Intra-class distance is displayed for five conditions: Enrollment temperature and voltage (E), usually $25\,^{\circ}C$*; Temperature decrease (T-), usually* $-40\,^{\circ}C$*; Temperature increase (T+), usually* $85\,^{\circ}C$*; Voltage decrease (V-), usually -10% Vdd; Voltage increase (V+), usually +10%. The next column denotes the average inter-device Hamming distance. Then, the min-entropy per bit and finally the CTW compression ratio is displayed, where less than 100% indicates compression occurred which is an indication of reduced entropy.*
When a paper not explicitly mentioned a performance figure, but provided a graphic or plot from which it could be deduced, an approximation sign is added (≈). The results from the latch proposal paper have been omitted from this table because the analysis methods used there are too different. Furthermore, the authors proposing the ring oscillator PUF only provided the worst-case intra-chip variation, which was below 1%.

Chapter 5

Framework for entropy estimation

In this chapter, before starting actual entropy estimations, a framework is introduced which models and formalizes the PUF creation and usage process. In this framework, all the relevant stochastic variables, model parameters and entropies are discussed and assigned a notation, providing a clear and consistent environment for the entropy calculations of the next chapter.

Although the focus lies on memory based PUFs, the framework can be used for any PUF which provides a bit string as output. For convenience, in this chapter the term 'cells' is used to denote the source of a PUF response, but this framework applies just as well to latches, flip-flops or other PUF types that provide a binary response.

5.1 Modelling a binary output PUF

A (memory based) PUF consists of a number of cells, which when measured, return a zero or one. When a PUF is created, every cell 'gains' a certain probability of starting up in the one state, a bias. Over all cells, we can view this process as drawing a vector of biases from a certain distribution.

A PUF can thus be mathematically represented by a vector of biases, where each element represents a cell's preference to start-up in the one state. The PUF creation process can be seen as drawing this set of biases from a large distribution. The entropy of this distribution determines the number of unique PUFs that can be created, and is known as the *creation entropy*.

However, not all information contained can be extracted in practice, due to the way PUFs are implemented in practice and due to noise. Therefore, creation entropy provides an upper bound on the entropy but is not the most useful measure. A better indication of uniqueness is the entropy that is left after processing, the *extractable entropy*. The extractable entropy bounds the maximal size of a cryptographic key that can be extracted from the PUF and the maximum number of PUFs that can be reliably distinguished in practice, e.g. used for identification.

Definition 5.1.1 *The extractable entropy is the entropy that is left after processing; the mutual information between enrollment and reconstruction.*

Processing consist of two phases, enrollment and reconstruction. The number of measurements done during these phases determines the precision with which the PUF bias can be estimated, the resolution. This resolution influences the entropy that can be extracted. Also, measurements done during reconstruction may occur in a different environment and may therefore suffer from increased noise.

Due to processing, the observed bias of a cell is turned into a discrete value, but the real bias is a continuous value. If one was able to measure this bias perfectly, a single cell would be enough to distinguish an infinite number of PUFs. However, due to practical limitations one can only obtain a bias estimate with limited resolution. Still, the number of possible sets of biases, the cardinality of the distribution from where the bias vectors are drawn, is enormous. This makes calculating entropy empirically impossible, since one has to know the probability that each possibility, a vector of certain biases, occurs. In other words, this requires the exact response distribution [49]. This is numerically infeasible to calculate due to the sheer number of probabilities that need to be known.

To get a feeling, suppose one is able to distinguish PUF biases with a resolution of m steps, e.g. by performing $m-1$ measurements resulting in a number between 0 and $m-1$. Then, the possible number of bias vectors for an n cell PUF is m^n. It is not unreasonable to do 40 measurements on a 8192 bit PUF, which spans an incalculable space. Besides, it is not uncommon to have only 20 samples available, which makes it hard to do any solid statements about a distribution that is so much larger. Thus, determining this entropy needs to be done in another way, which is one of the goals of this thesis and handled in the next chapter.

5.2 Formalism

5.2.1 Creation

Consider n cells with for each cell $i \in \{1, \ldots, n\}$ a **bias** $b_i \in [0,1]$ that denotes the preference of to start up in a zero or one state.

$$\text{Start-up value cell } i = \begin{cases} 1 & \text{with probability } b_i \\ 0 & \text{with probability } 1 - b_i. \end{cases}$$

The biases of all the PUFs cells can be represented in a vector of size n:

$$\vec{b} = \begin{pmatrix} b_1 \\ \ldots \\ b_n \end{pmatrix}.$$

In general, these biases depend on the non-deterministic creation process and as such can be modelled as continuous *random variable* $\vec{B}$ with probability density function ρ.

$$\vec{B} \sim \rho$$
$$\rho : [0,1]^n \to [0,\infty)$$

Throughout this thesis, the function mapping a bias vector to probabilities is labelled the **bias distribution**. The entropy of this distribution ρ, denoted $H(\vec{B})$, is the **creation entropy**. In theory this entropy is infinite since $\vec{B}$ is continuous but in practice we will work with a discretization of $\vec{B}$, limiting the entropy. The remaining entropy provides an upper bound on how many devices can be distinguished. It is closely related to the inter-device distance, since it determines the uniqueness between created devices. If creation entropy is low the number of uniquely distinguishable PUFs is also low.

5.2.2 Processing

PUF response processing consists of two steps, enrollment and reconstruction.

- Enrollment

  During enrollment, a PUF is challenged k times and the measured response is recorded. Here we define x_i as the number of times a '1' was recorded as start-up value of cell i.

  Thus we have that $x_i \in \{0, \ldots, k\}$ and discrete random variable $\vec{X} = (x_1, \ldots, x_n)$.

- Reconstruction

  During reconstruction, a PUF is challenged l times and the measured response is recorded. Here we define y_i as the number of times a '1' was recorded as start-up value of cell i. Thus we have that $y_i \in \{0, \ldots, l\}$ and discrete random variable $\vec{Y} = (y_1, \ldots, y_n)$. Often with reconstruction, $l = 1$, in which case y_i is the recorded response of the ith cell which is either 0 or 1.

During these steps, the bias of a cell may differ. Therefore, the bias under enrollment conditions is denoted b_{e} while the bias under reconstruction conditions is written as b_{r}.

These two processes determine the resolution with which PUF biases can be determined. It greatly depends on the number of measurements during enrollment (k) and reconstruction (l) how many PUFs can be distinguished in practice, and thus, what the extractable entropy is.

The extractable entropy can be seen as the information that is present in the enrollment measurements as well as in the reconstruction measurements, the *mutual information*. Mutual information is a measure of the amount of information a random variable shares with another random variable and is denoted as $I(\vec{X}; \vec{Y})$. In the context of the here described enrollment and reconstruction processes, mutual information tells us how much uncertainty about $\vec{Y}$ is reduced when one knows $\vec{X}$. Stated differently, it quantifies the uncertainty that is left after knowing the remaining entropy of reconstruction given enrollment and vice versa. As such it provides a measure for both uniqueness and reliability of a PUF.

Intuitively, low mutual information indicates weak correlation between the reconstructed response and the enrolled response making it hard to reliably authenticate a PUF response. Low mutual information also indicates a small output space for the fuzzy extractor which negatively affects uniqueness. Furthermore, if we want to extract an s-bit secret from a PUF, we can never do better than $I(\vec{X}; \vec{Y})$, thus $s \leq I(\vec{X}; \vec{Y})$.

Chapter 6

Entropy estimation

Now that a proper theoretic framework is in place, it is time to start with entropy estimation methods. Estimating the creation entropy is done by determining the entropy of the bias distribution, i.e. $H(B)$. To calculate this entropy, it is required to know the bias distribution ρ.

Furthermore, the main research goal is to come up with a method to estimate the extractable entropy. This extractable entropy can be found by calculating the mutual information between the enrollment and reconstruction measurements. However, this also requires knowledge of the bias distribution ρ (during both enrollment and reconstruction) in order to calculate the required entropies. Therefore, this chapter starts out by describing techniques to estimate the bias probability distribution.

6.1 Estimating the distribution of biases using the maximum entropy principle

With only a few samples, it is statistically not sound to determine the bias distribution (ρ) empirically. Nevertheless, to calculate an entropy estimate a bias probability distribution is required. In order to estimate this probability distribution, it is best to minimize assumptions and introduced biases. In other words, the most optimal probability distribution is the one that has the largest remaining uncertainty and thus the maximum entropy, which will provide an upper bound. This is known as the principle of maximum entropy [40].

In the PUF context, this means that with only a few samples it is possible to determine a mean and variance, and based on these parameters, a bias probability distribution can be estimated.

For given mean and variance, the (creation) entropy is maximal when the vector $\vec{X}$ has a multivariate normal distribution ($\vec{X} \sim \mathcal{N}(\vec{\mu}, \Sigma)$) with probability density function

$$f_{\vec{X}}(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu})^{\mathrm{T}} \Sigma^{-1} (\vec{x}-\vec{\mu})} \tag{6.1}$$

having expectation

$$\vec{\mu} = (\mathbb{E}[X_1], \ldots, \mathbb{E}[X_n]) \tag{6.2}$$

and $n \times n$ covariance matrix

$$\Sigma_{ij} = \mathrm{Cov}[X_i, X_j] \text{ with } i, j = 1 \ldots n. \tag{6.3}$$

The differential entropy can then very easily be calculated by ([3, 40]):

$$h(f) = \frac{1}{2}\ln\Big((2\pi e)^n|\Sigma|\Big). \tag{6.4}$$

However, this only holds if the variables lie in the infinite domain, which is not the case for our bias vectors. Thus, we need to find a probability density function that has maximal entropy on the domain $[0,1]$ and a method to determine the entropy of this new probability density function. We argue that, for the simplified univariate case, i.e. considering a single cell, the maximum entropy bias distribution has a Gaussian form, specifically the following:

$$\rho = e^{-\lambda_2 b^2 + \lambda_1 b}. \tag{6.5}$$

The proof of this as well as the relation between sample estimates μ, σ^2 and distribution parameters λ_1, λ_2 is given in Appendix A.

Thus, by considering a sample set of PUF data one can extract the mean and variance, and by expressing them in terms of λ_1 and λ_2, an upper bound on the bias probability distribution ρ can be determined. This distribution can then be used to estimate an upper bound on the entropy.

6.1.1 Estimating bias probability distribution empirically

If more samples are available, it is better to base the bias distribution on histograms obtained from empirical data. In this approach, a probability mass function with a fixed number of bins is created. Each bin represents the probability to start in the one state, thus a discrete bias.

For each cell, the weights of the bins are determined by measuring a sample collection of PUFs multiple times and counting the occurrences of a specific number of times a cell started in the one state. This means that the number of bins should always be less than the number of measurements or PUFs used.

6.2 Estimating the extractable entropy

When a bias probability distribution is available, it is possible to calculate entropies. In order to determine the extractable entropy, we need to determine the mutual information between enrollment and reconstruction measurements. First, the mutual information for a single cell i is considered. Note that, for clarity, the subscript i has been omitted until the method is generalized to multiple cells:

$$I(X;Y) = H(X) + H(Y) - H(X,Y). \tag{6.6}$$

In order to calculate these entropies, the probabilities p_x, q_y and r_{xy} are required. These respectively denote the probability of x cells starting in the one state during enrollment (out of k), y cells starting in the one state during reconstruction (out of l) and the joint probability of both. In this section, the bias distribution function is considered to be continuous, the discrete version of the analysis is given in the next subsection.

These probabilities, with here p_x as example, can be calculated using joint density function $(B_\mathrm{e}, X) \sim \mathbb{P}$ and marginalising out B. In a similar fashion, q_y can be calculated using $(B_\mathrm{r}, Y) \sim \mathbb{Q}$. These joint probability functions can be rewritten using the chain rule, which means the conditional probabilities $p_{x|b_\mathrm{e}}$ and $q_{y|b_\mathrm{r}}$ are required. Here, b_e and b_r stand for the bias under enrollment and reconstruction conditions respectively.

Since these conditional probabilities are binomially distributed, we have:

$$p_{x|b_\mathrm{e}} = \binom{k}{x} b_\mathrm{e}^x (1-b_\mathrm{e})^{k-x}. \tag{6.7}$$

$$q_{y|b_\mathrm{r}} = \binom{l}{y} b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y}. \tag{6.8}$$

Using these, the probabilities p_x, q_y and r_{xy} can be expressed as:

$$\begin{aligned} p_x &= \int_0^1 \mathrm{d}b_\mathrm{e}\, \mathbb{P}(b_\mathrm{e}, x) = \int_0^1 \mathrm{d}b_\mathrm{e}\, p_{x|b_\mathrm{e}}\, \rho_\mathrm{e}(b_\mathrm{e}) \\ &= \binom{k}{x} \int_0^1 b_\mathrm{e}^x (1-b_\mathrm{e})^{k-x} \rho_\mathrm{e}(b_\mathrm{e})\, \mathrm{d}b_\mathrm{e}. \end{aligned} \tag{6.9}$$

$$\begin{aligned} q_y &= \int_0^1 \mathrm{d}b_\mathrm{r}\, \mathbb{Q}(b_\mathrm{r}, y) = \int_0^1 \mathrm{d}b_\mathrm{r}\, q_{y|b_\mathrm{r}}\, \rho_\mathrm{r}(b_\mathrm{r}) \\ &= \binom{l}{y} \int_0^1 b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y} \rho_\mathrm{r}(b_\mathrm{r})\, \mathrm{d}b_\mathrm{r}. \end{aligned} \tag{6.10}$$

$$r_{xy} = \int_0^1 \mathrm{d}b_\mathrm{e}\, \rho_\mathrm{e}(b_\mathrm{e}) p_{x|b_\mathrm{e}} \int_0^1 \mathrm{d}b_\mathrm{r}\, \rho_{r|e}(b_\mathrm{r}|b_\mathrm{e}) q_{y|b_\mathrm{r}}. \tag{6.11}$$

Next we generalize this for multiple cells using the earlier introduced vectors:

$$I(\vec{X}, \vec{Y}) = H(\vec{X}) + H(\vec{Y}) - H(\vec{X}, \vec{Y}). \tag{6.12a}$$

$$p_{\vec{x}} = \left[\prod_{j=1}^n \binom{k}{x_j} \right] \cdot \int_{[0,1]^n} \mathrm{d}^n b_\mathrm{e}\, \rho_\mathrm{e}(\vec{b}_\mathrm{e}) \left[\prod_{i=1}^n b_{\mathrm{e}i}^{x_i} (1-b_{\mathrm{e}i})^{k-x_i} \right]. \tag{6.12b}$$

$$q_{\vec{y}} = \left[\prod_{j=1}^n \binom{l}{y_j} \right] \cdot \int_{[0,1]^n} \mathrm{d}^n b_\mathrm{r}\, \rho_\mathrm{r}(\vec{b}_\mathrm{r}) \left[\prod_{i=1}^n b_{\mathrm{r}i}^{y_i} (1-b_{\mathrm{r}i})^{l-y_i} \right]. \tag{6.12c}$$

$$r_{\vec{x}\vec{y}} = \int_{[0,1]^n} \mathrm{d}^n b_\mathrm{e}\, \rho_\mathrm{e}(\vec{b}_\mathrm{e}) p_{\vec{x}|\vec{b}_\mathrm{e}} \int_{[0,1]^n} \mathrm{d}^n b_\mathrm{r}\, \rho_{\mathrm{r}|\mathrm{e}}(\vec{b}_\mathrm{r}|\vec{b}_\mathrm{e}) q_{\vec{y}|\vec{b}_\mathrm{r}}. \tag{6.12d}$$

Here, the conditional probabilities factorise as:

$$p_{\vec{x}|\vec{b}_\mathrm{e}} = \prod_{i=1}^n p_{x_i|b_{\mathrm{e}i}}. \tag{6.13}$$

$$q_{\vec{y}|\vec{b}_\mathrm{r}} = \prod_{i=1}^n q_{y_i|b_{\mathrm{r}i}}. \tag{6.14}$$

6.2.1 Simplification

Calculating the mutual information using the multivariate approach with continuous probability distributions is very difficult in practice. Therefore, a simplification is given to determine the extractable entropy using a discrete probability distribution. In this case, the cells are assumed to have independent bias distributions. This approach will also be used later for the entropy analysis of the PUF data.

In order to avoid confusion with the continuous case, we denote the discrete bias probability mass function τ. For each cell, the probabilities necessary for calculating the mutual information can be calculated as (again, the cell index i has been omitted for clarity):

$$p_x = \binom{k}{x} \sum_{b_\mathrm{e} \in \mathrm{bins}_\mathrm{e}} b_\mathrm{e}^x (1-b_\mathrm{e})^{k-x} \tau_\mathrm{e}(b_\mathrm{e}). \tag{6.15a}$$

$$q_y = \binom{l}{y} \sum_{b_\mathrm{r} \in \mathrm{bins}_\mathrm{r}} b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y} \tau_\mathrm{r}(b_\mathrm{r}). \tag{6.15b}$$

$$r_{yx} = \binom{k}{x}\binom{l}{y} \sum_{b_\mathrm{e} \in \mathrm{bins}_\mathrm{e}} b_\mathrm{e}^x (1-b_\mathrm{e})^{k-x} \tau_\mathrm{e}(b_\mathrm{e}) \sum_{b_\mathrm{r} \in \mathrm{bins}_\mathrm{r}} b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y} \tau_{\mathrm{r}|\mathrm{e}}(b_\mathrm{r}|b_\mathrm{e}). \tag{6.15c}$$

Using these probabilities, we can calculate $H(X_i)$, $H(Y_i)$ and $H(X_i, Y_i)$ for every cell and thus obtain the total mutual information as $I(\vec{X};\vec{Y}) = \sum_{i=1}^{n} I(X_i;Y_i)$ under the assumption that cells are independent.

There are two potential difficulties with using this approach, specifically with the term $\tau_{r|e}$ which models the noise. First, the number of devices must be significantly larger than the number of distinguishable biases during the enrollment and reconstruction phase, otherwise it is not possible to estimate an accurate noise model. Second, calculating a noise model per cell is computationally expensive. Therefore, another simplification is presented using a global noise model.

The calculation for p_x remains unchanged, but in the calculation of q_y the bias distribution of the reconstruction condition is no longer sampled empirically but estimated using individual enrollment bias distributions and the global noise model. This results in the following equation for q_{yi} for cell i:

$$q_{yi} = \binom{l}{y} \sum_{b_\mathrm{r} \in \mathrm{bins}_\mathrm{r}} \left(\sum_{b_\mathrm{e} \in \mathrm{bins}_\mathrm{e}} \tau_{\mathrm{e}i}(b_\mathrm{e}) \tau_{\mathrm{r}|\mathrm{e}}(b_\mathrm{r}|b_\mathrm{e}) \right) b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y}. \tag{6.16}$$

In a similar fashion, r_{xyi} is calculated using an individual τ_{ei} per cell but a global noise model:

$$r_{yxi} = \binom{k}{x}\binom{l}{y} \sum_{b_\mathrm{e} \in \mathrm{bins}_\mathrm{e}} b_\mathrm{e}^x (1-b_\mathrm{e})^{k-x} \tau_{\mathrm{e}i}(b_\mathrm{e}) \sum_{b_\mathrm{r} \in \mathrm{bins}_\mathrm{r}} b_\mathrm{r}^y (1-b_\mathrm{r})^{l-y} \tau_{\mathrm{r}|\mathrm{e}}(b_\mathrm{r}|b_\mathrm{e}). \tag{6.17}$$

This solution is not optimal, but preferable when there is not enough data to form an accurate noise model for each cell.

6.3 Entropy estimation based on degrees of freedom in binomial fit (Daugman)

Here, a modification of the iris recognition algorithm of Daugman (introduced in Section 4.2 on previous work) is introduced to estimate creation entropy. Instead of comparing Hamming distances of (binary) responses, the Hamming weights per cell are compared. Let the Hamming weight of the string of start-up values for cell i and PUF p be denoted as defined in the framework, $x_i^p \in 0..k$. Then the distance between two PUF cells can calculated as:

$$d(p, p', i) = |x_i^p - x_i^{p'}|. \tag{6.18}$$

Although the Hamming weight of a string of start-up values of a cell is binomially distributed, the distance between two Hamming weights is not. However, to adopt the method by Daugman we pretend as if the resulting distances are binomially distributed.

To obtain the total difference between two PUFs, Equation 6.18 is repetitively applied to all PUF cells. By summing the resulting cell differences and by dividing by the number of cells n and the number of measurements k, a fractional distance is obtained denoting the average difference in biases between these PUFs.

By pairwise comparing all PUFs,

$$\mathcal{D} = \{\frac{1}{nk}\sum_{i=1}^{n} d(p, p', i) : p, p' \in \{1, \ldots, P\},\ p \neq p'\}, \tag{6.19}$$

a histogram of the distribution of distances can be created. By scaling the distances to denote the average bias distance (ratio), a fractional binomial can be fitted through the resulting distribution which provides an indication of uniqueness.

This fractional binomial has on the x-axis the ratio that two PUFs differ, and on the y-axis the probability of this happening (given that two PUFs are indeed distinct). With $x \in [0, 1]$, this fit is defined as:

$$\text{Fit}(N, x, p) = \binom{N}{xN} p^{xN}(1-p)^{N-xN}. \tag{6.20}$$

The number of trials N and the probability of success p can be derived from the data. It holds that the mean is given by pN and the variance by $Np(1-p)$. Because the fractional binomial is scaled by N, the mean is just equal to p and the variance can be calculated as $p(1-p)/N$. With the parameters N and p, the creation entropy can be calculated as:

$$H(\vec{B}_e) = N \cdot H(p). \tag{6.21}$$

Chapter 7

Evaluation of entropy estimation methods

The goal of this chapter is to analyse the performance of both the entropy estimation methods found in the literature and those introduced in the previous chapter. In order to do this, the methods are benchmarked against a known synthetic distribution.

7.1 Benchmarking entropy estimation methods against synthetic data

For the entropy estimation method assessment, we mimic the PUF creation process by creating a synthetic bias probability distribution. Because the actual entropy of this synthetic distribution is known, the performance of the entropy estimation methods can be compared. Furthermore, this enables us to compare all methods using the same dataset.

Since we later apply these estimation methods to the UNIQUE PUF data, the parameters for the synthetic bias probability distribution are obtained by estimating the bias probability distribution of real PUF data. In Section 6.1, two methods for this goal are provided: the maximum entropy method yielding a continuous probability density function as an upper bound on the real bias distribution, and a discrete probability mass function estimated from sample histograms.

We argue here that the first method is totally unsuitable for (our) binary output PUF data. Therefore, throughout the entire analysis of both synthetic and actual PUF data the bias distributions will be estimated using the second method. The reason for this is that the actual PUF data shows that biases are not at all Gaussian-distributed. By plotting the observed biases, it is clear that the bias distribution has almost all of the probability concentrated near the edges, which is depicted in Figure 7.1b. Since the maximum entropy estimation method provides such a poor match for the actual bias distribution (as shown in Figure 7.1a), the method is disregarded in this thesis. However, for other PUFs that have a more diverse bias, for example one that has the probability of a cell to start-up as a one concentrated near 0.5, the maximum entropy estimation method could provide an accurate estimation for the actual bias probability distribution. Possibly, the maximum entropy estimation method could be used to estimate an upper bound on the entropy of true random number generators, since cells with biases that are Gaussian distributed around 0.5 would be ideal candidates for generating true random numbers.

7.1.1 Approach of synthetic benchmark

To evaluate the uniqueness measures and entropy estimation method, a step-by-step approach is taken to set up a synthetic benchmark. Here these steps are detailed, and in the next section they are executed.

1. Determine form and parameters for the synthetic probability distribution

   To decide on the form of the synthetic distribution, real PUF data is examined. Per cell, a histogram is created with biases on the x-axis and the count of how often PUFs have this bias on the y-axis. By inspecting this histogram, the form and parameters of the most likely distribution can be determined.

2. Create bias probability distribution

   Once the form and parameters are established the actual cell bias probability distribution is created. This is a discrete probability mass function with probabilities associated to bins denoting a certain bias. By using the Shannon entropy formula on the probabilities (Equation 2.2), the creation entropy can be calculated.

3. Draw enrollment PUFs from these distribution

   From these bias probability distributions, denoted τ_{ei}, we can draw the preference of a simulated cell to start-up in the one-state. This is repeated n times where n denotes the number of cells depending on the PUF type we wish to simulate. The result is a synthetic PUF in an enrollment environment, which can be represented as a vector of biases $\vec{b}_e$.

4. Sample data from enrollment PUFs

   Sample data is created by repeatedly (for all n cells) performing Bernoulli trials where success denotes the cell started up in the one state. The probability of success is determined by the bias vector $\vec{b}_e$.

5. Transform enrollment PUFs to reconstruction PUFs using a transformation matrix

   To simulate how PUF biases change in the reconstruction phase, we again resort to actual PUF data. For each cell of the actual PUF data, we look at which bias bin it is in during enrollment measurements and again during reconstruction measurements.

   More formally, we form the $l_e \times l_r$ transformation matrix T where l_e and l_r denote the number of bins used during enrollment and reconstruction respectively. Rows in T corresponds to bins in the enrollment phase and columns to bins in the reconstruction phase. The cells of T count how often a bias change is observed, so $T(1,2) = 3$ means that 3 cells that during enrollment ended up in bin 1 (had the lowest bias) during reconstruction changed to bin 2. From this matrix, the probabilities $p_{y|x}$ can be derived by transforming the columns into probability distributions. In other words, the transformation matrix now represents the global noise model $\tau_{r|e}$.

   Based on this transformation matrix, it is possible to transform an enrollment PUF $\vec{b}_e$ to a reconstruction PUF $\vec{b}_r$. After this step, the mutual information can be calculated giving the actual extractable entropy.

6. Sample reconstruction PUFs

The vector of biases $\vec{b}_r$ is used to sample reconstruction measurements from. This process is identical to step 4.

7. Calculate and compare entropy

   Now that we have a distribution of known entropy, enrollment samples and reconstruction samples, it is possible to apply the entropy estimation methods to the samples and compare the results to the known entropy.

7.2 Assessment of entropy estimation methods

Before the approach described above can be followed, first, a decision has to be made which PUF type will serve as inspiration for the synthetic data. We choose to determine the form and parameters of the synthetic bias probability distribution based on the DFF PUF data. This choice is based on the fact that most PUF types present have some biasing, i.e. the probability of starting up in the one or zero state is not equal, but is otherwise arbitrary.

Histograms of PUF start-up behaviour, illustrating that most PUF types suffer from some biasing, are given in Figure 7.2. The data displayed in the histograms[1] is obtained using 192 devices with 60 enrollment measurements at room temperature and 40 reconstruction measurements at either -40 or $+85$ °C.

These histograms show that cells have a very strong bias, which means they favour a one or zero outcome strongly as can be seen by high counts near the edges of the histogram. This holds for all the (memory based) PUF types encountered on the UNIQUE ASIC. However, the balance of how many cells favour a bias of 0 or 1 strongly differs across types and also depends on the reconstruction condition.

Furthermore, it is worthwhile to note that reconstruction conditions effect the biasing differently for particular PUF types. When the temperature rises, the DFF PUF increases in performance (suffers from less biasing) while the latch PUFs performance decreases. In either case, the number of 'ones' in a PUF response depends strongly on the environmental temperature. The SRAM PUF however seems almost unaffected by temperature changes.

To decide on the form of the synthetic distribution, the bias histogram of Figure 7.2 is inspected. This figure shows the average bias distribution over all cells. Most of the time, the cells of a DFF PUF start 60 out of 60 times in the one state, and besides the other peak at 0, the histogram suggests that it is uncommon to have cells that have a weak or no preference regarding start-up state.

Since the histogram shows no similarity to known distributions, a categorical distribution is chosen with bins denoting the probability of starting in the one state. Based on this form, a discrete probability distribution is created which **from here on is treated as the real bias probability distribution under enrollment, denoted** τ_e. As the DFF data consists of sixty measurements, our probability distribution has sixty-one distinguishable biases $(0, \frac{1}{60}, \ldots, 1)$.

To determine the probabilities of a bias distribution of an individual cell, a histogram is created of the number of measurements per PUF that resulted in one-bit. An example is depicted in depicted in Figure 7.3. Thus, the x-axis contains a count from 0 to 60 denoting the number of ones during enrollment and the y-axis denotes how often this occurs, i.e. the number of PUFs

[1] An extended description of the dataset can be found in the next chapter where the entire UNIQUE dataset is described

#Bins:	2	3	4	5	6	11	21	41	**61**
Creation entropy:	0.87	1.02	1.11	1.18	1.24	1.41	1.60	1.73	**1.90**

Table 7.1: Table denoting creation entropy which increases with the number of distinguishable bins (biases).

having this count. By dividing the values on the x-axis by 60 and scaling the y-axis by the total number of PUFs, the probability for a cell to start-up in the one state — the bias — is obtained (earlier displayed in Figure 7.1b). This process is repeated for each cell to create individual bias distributions.

By applying the Shannon entropy formula to the probabilities of these bias distributions, the maximum uncertainty in bits per cell can be calculated; the creation entropy. This entropy is never achieved in reality due to noise and is also dependent on the number of distinguishable biases, the resolution. If the resolution is decreased, the entropy also decreases. This is displayed in Table 7.1.

This table already hints that increasing the number of measurements of a cell during enrollment increases entropy, albeit with diminishing effectiveness: only the first few extra measurements have a relatively large effect.

Using the bias distributions $\tau_{\mathrm{e}i}$, synthetic PUFs are created. For every PUF, 8192 cells are simulated by drawing for each cell i a bias from this distribution. For this synthetic benchmark, 192 synthetic PUFs are created using vectors of 8192 biases. These vectors are stored in a matrix of biases $b^p_{\mathrm{e}i}$ where subscript i is the cell index and superscript p denotes the synthetic PUF number (out of $P = 192$):

$$\text{Synthetic PUF enrollment matrix} = \begin{pmatrix} b^1_{\mathrm{e}1}, b^1_{\mathrm{e}2} \dots b^1_{\mathrm{e}n} \\ b^2_{\mathrm{e}1}, b^2_{\mathrm{e}2} \dots b^2_{\mathrm{e}n} \\ \dots \\ b^P_{\mathrm{e}1}, b^P_{\mathrm{e}2} \dots b^P_{\mathrm{e}n} \end{pmatrix}. \tag{7.1}$$

To be able to calculate the extractable entropy, synthetic reconstruction PUFs are needed since a PUF behaves differently (noisy) under environmental changes.

To create these synthetic reconstruction PUFs, the change across operation conditions is inspected of real PUF data, where the reconstruction data consists of measurements of the same DFF PUF devices but changing the environment temperature to +85 °C.

Because not enough data is available to accurately model the change of individual cells, a global transformation matrix is created with on the rows the possible values of the bias under enrollment conditions (b_e) and on the columns the bias values under reconstruction conditions (b_r). The values in the matrix denote the probability of a cell to have bias b_r under reconstruction conditions, given that the cell had bias b_e during enrollment, thus $p(b_r|b_e)$. This is simply achieved by counting actual bin changes from experimental data. For the synthetic analysis, this transformation matrix is assumed to perfectly model the actual change to the bias of enrollment PUFs when brought into reconstruction conditions.

The result of this procedure is displayed in Figure 7.4a, which mainly shows that most cells that start only zero out of sixty times in the one-state during enrollment remain to do so under reconstruction conditions, but some cells that start sixty out of sixty times during enrollment in the one-state, suddenly start zero times in the one-state during reconstruction.

Using this matrix, the earlier created enrollment PUFs can be 'transformed' to reconstruction

PUFs by considering, for each cell, the probability of that cells' bias to change under reconstruction conditions given the bias it currently has, which is displayed on the y-axis in Figure 7.4b. This probability is denoted in the rows of Figure 7.4b, which illustrates that it is very likely to have a bias close to zero or one during reconstruction, even if the bias was not near the edges during enrollment.

This can be explained by the fact that it is very unlikely to have such a bias under any condition, thus even if a few cells have a bias not equal to zero or one, there is a massive amount of cells having a bias near the edges. Note that the height in Figure 7.4a is displayed on a logarithmic scale, the values in the inner region are very low compared to the peaks in the corners.

The resulting reconstruction PUFs are stored in a matrix of biases $b^p_{\mathrm{r}i}$ where subscript i is the cell index and superscript p denotes the synthetic PUF number (out of P):

$$\text{Synthetic PUF reconstruction matrix} = \begin{pmatrix} b^1_{\mathrm{r}1}, b^1_{\mathrm{r}2} \dots b^1_{\mathrm{r}n} \\ b^2_{\mathrm{r}1}, b^2_{\mathrm{r}2} \dots b^2_{\mathrm{r}n} \\ \dots \\ b^P_{\mathrm{r}1}, b^P_{\mathrm{r}2} \dots b^P_{\mathrm{r}n} \end{pmatrix}. \tag{7.2}$$

Now that the transformation matrix and both enrollment and reconstruction PUFs are known, it is possible to calculate the theoretical maximum mutual information between enrollment and reconstruction by considering the procedure denoted in Section 6.2.1, which is explained in more detail in the next subsection.

With both synthetic enrollment PUFs and reconstruction PUFs available, it is possible to create synthetic measurements which are just binary strings denoting the cells power-up value. From each synthetic PUF, 60 enrollment measurements are taken by performing a Bernoulli trial for each cell (row in matrix 7.1), where the probability of success is given by the cells' bias. Success means the PUF started up in the one state during this measurement. This procedure is repeated to create reconstruction data using matrix 7.2, only now with 40 measurements.

7.2.1 Evaluation

Using this synthetic data, it is possible to assess the entropy estimation methods. Here the actual results are presented as obtained by performing the uniqueness and entropy estimation methods described in previous chapters.

Properties of synthetic data

First, a recap is given of the maximum achievable creation entropy as contained by the enrollment bias probability distribution for different resolutions (in Table 7.1). This time, also the improvement and relative improvement of the entropy as the resolution increases is displayed. As soon as more measurements are taken, more options are possible and thus the entropy slightly rises, although this effect diminishes quickly. This is because the distribution is concentrated near the edges and has little weight in the middle.

Second, the maximum achievable mutual information is calculated using the bias probability distributions used to create synthetic PUFs and the global transformation matrix. To do this, three entropies need to be determined using Equation 2.2: $H(X), H(Y), H(X,Y)$. To calculate $H(X)$, only the probability of starting x out of k enrollment measurements in the one-state is needed, p_x. This probability can be calculated using Equation 6.15a. Here, τ_{e} is given by the bias probability distribution, and b_{e} is taken from the 61 possible biases (0–60).

#Bins	2	3	4	5	6	11	21	41	61
Creation entropy:	0.87	1.02	1.11	1.18	1.24	1.41	1.60	1.73	1.90
Improvement	-	0.15	0.09	0.07	0.06	0.17	0.19	0.13	0.17
Relative improvement	-	0.15	0.09	0.07	0.06	0.03	0.02	0.01	0.01

Table 7.2: Table of creation entropy

Now, computing $H(Y)$ and $H(X,Y)$ is slightly more involved. For this, we need q_y and r_{xy} which both can be calculated using the noise model. To calculate $H(X,Y)$, the joint probability of x and y cells starting as a one (during enrollment and reconstruction respectively) is needed. This probability is calculated using Equation 6.17 where τ_{e} is the bias probability distribution of the enrollment condition and $\tau_{\mathrm{r|e}}$ is the global noise model. Finally, $H(Y)$ is calculated using probability q_y. As we consider a global noise model, this probability can be calculated using Equation 6.16.

With these three entropies, the mutual information $I(X;Y) = H(X) + H(Y) - H(X,Y)$ is calculated. This results in a maximal achievable mutual information of 0.46 bits per cell.

7.2.2 Assessment of entropy estimation methods

Here, the methods to obtain entropy estimations detailed in previous chapters are discussed. Three methods to calculate an estimate of the creation entropy are discussed: The min-entropy estimation method, which obtains a lower bound on the creation entropy, CTW compression test and the Daugman method. Of these methods, the CTW compression test provides an estimate for the creation entropy given one enrollment measurement (per PUF), while the Daugman and min-entropy methods allow to estimate entropy based on a variable number of enrollment measurements. Lastly, the mutual information is calculated to provide an estimate of the extractable entropy.

Every estimation method is applied to the full dataset of 192 synthetic PUFs and to a subset of only 20 PUFs. The goal of this is to determine the influence of the number of PUFs on the estimation accuracy.

Min-entropy The min-entropy method as handled in the section on previous work is slightly modified to work with multiple enrollment measurements. The min-entropy is calculated by predicting for each cell a bias distribution and determining the most likely bias (bias with highest probability), i.e. the highest peak in the histogram of the distribution. This probability is then fed to Equation 2.4 to calculate the min-entropy. The final min-entropy per cell is obtained by averaging the min-entropy over all cells, which is displayed in Table 7.3. Additionally, the min-entropy calculated using a single measurement is given.

P	192	20
$k = 60$	0.67	0.68
$k = 1$	0.52	0.52

Table 7.3: Min-entropy estimation for 192 and 20 synthetic PUFs.

CTW Compression For the inter-device CTW compression test, measurements from different devices are concatenated. The resulting bit string should not be compressible. If compression occurs, this means the different devices have some correlation, indicating a lack of entropy. This is the case for our synthetic PUFs, since the CTW algorithm is able to compress the responses to 88% of their original length. The compression test achieves this ratio for both 20 and 192 devices used.

In addition, an intra-device compression test — where measurements from the same synthetic PUF are concatenated — is performed. The resulting string should be compressible provided the PUF is reliable, which is the case as the CTW algorithm is able to compress intra-device responses to 17% of its original length.

Daugman-inspired method For each cell, the difference in estimated bias between synthetic PUFs is pairwise compared. These differences are averaged to obtain the average distance between two PUFs. For the analysis, 20 and 192 synthetic PUFs are used, which means that respectively 190 and 18336 comparisons and thus distances are obtained. From these distances, a histogram is formed through which a binomial distribution is fitted. This approach is taken for several different numbers of enrollment measurements. The parameters of the binomial fit are displayed in Table 7.4. Here, k denotes the number of enrollment measurements used, N the number of trials and p the success probability in each trial. For various values of k the histogram and fit is plotted in Figure 7.5.

k	N	p	$N \cdot h(p)$	Entropy per cell
1	8162	0.418	8001.488	0.977
2	8687	0.417	8514.334	1.039
3	9116	0.417	8932.381	1.090
5	9197	0.416	9009.943	1.100
10	9390	0.416	9198.365	1.123
20	9177	0.416	8988.750	1.097
30	9856	0.416	9653.230	1.178
60	9483	0.416	9287.713	1.134
k	N	p	$N \cdot h(p)$	Entropy per cell
1*	7627	0.417	7473.678	0.912
2	8219	0.416	8051.348	0.983
3	8496	0.416	8321.361	1.016
5	8806	0.415	8623.603	1.053
10	8879	0.415	8693.714	1.061
20	9013	0.415	8824.017	1.077
30	9020	0.415	8830.536	1.078
60*	8997	0.415	8807.733	1.075

Table 7.4: Table showing the results of the analysis of our modified Daugman approach. The first table is created using 20 synthetic PUFs, while the second table uses 192 PUFs. The binomial fit of the entries with a star are plotted in Figure 7.5.

Mutual Information In order to calculate the mutual information, the data of synthetic enrollment PUFs are used to estimate the bias probability distribution of cells during enrollment,

τ_{ei}. This is done for several number of measurements k, hence obtaining bias distributions with different resolutions.

Then, the global noise model $\tau_{r|e}$ is created by observing the changes to the bias of cells from enrollment to reconstruction. Using the individual bias probability distributions τ_{ei} and the global noise model $\tau_{r|e}$, the probabilities p_x, q_y and r_{xy} are calculated using Equations 6.15a, 6.16 and 6.17. Then, using these probabilities the mutual information between enrollment and reconstruction measurements is calculated and displayed in Table 7.5. The table lists the mutual information for combinations of k and l. It is clear that the extractable entropy increases as the number of measurements during enrollment and reconstruction increases, with a maximum increase of 0.13 bit per cell.

k,l	1	2	3	5	10	20	40
1	0.31	0.33	0.33	0.34	0.35	0.35	0.36
2	0.33	0.35	0.35	0.36	0.37	0.37	0.38
3	0.34	0.35	0.36	0.37	0.38	0.38	0.39
5	0.35	0.36	0.37	0.38	0.39	0.40	0.40
10	0.36	0.38	0.38	0.39	0.40	0.41	0.41
20	0.37	0.38	0.39	0.40	0.41	0.42	0.42
40	0.37	0.39	0.40	0.41	0.42	0.42	0.43
60	0.37	0.39	0.40	0.41	0.42	0.43	0.43

k,l	1	2	3	5	10	20	40
1	0.32	0.34	0.34	0.35	0.36	0.36	0.37
2	0.34	0.36	0.36	0.37	0.38	0.39	0.39
3	0.35	0.37	0.37	0.38	0.39	0.40	0.40
5	0.36	0.38	0.39	0.39	0.40	0.41	0.41
10	0.37	0.39	0.40	0.41	0.42	0.42	0.43
20	0.38	0.40	0.41	0.42	0.43	0.43	0.44
40	0.38	0.40	0.41	0.42	0.43	0.44	0.44
60	0.39	0.41	0.42	0.42	0.44	0.44	0.45

Table 7.5: Estimation of extractable entropy in bits per cell of synthetic PUF data by calculating the mutual information for various values of k and l. The results of the first table are calculated using 20 devices, while the results of the second table are calculated using 192 devices.

7.2.3 Summary of evaluation results

The min-entropy test found the lowest entropy per cell, which comes as no surprise. Judging from Table 7.2, the calculated min-entropy value, which is sometimes used as a lower bound on creation entropy [8, 41], provides a substantial under-estimation. When using a single measurement per PUF to determine the min-entropy, the under-estimation lies around 0.35 bit per cell when compared to the Shannon entropy shown in Table 7.2. Whether 20 or 192 devices are used for this test does not matter as the results are almost the same[2].

The influence of the number of PUFs is also minimal for the CTW compression test. As the results for CTW compression did not change much by decreasing the number of synthetic PUFs used from 192 to 20, the compression test was repeated with only a single PUF. This already

[2]In the later analysis on the UNIQUE PUF data the number of devices does matter for this test

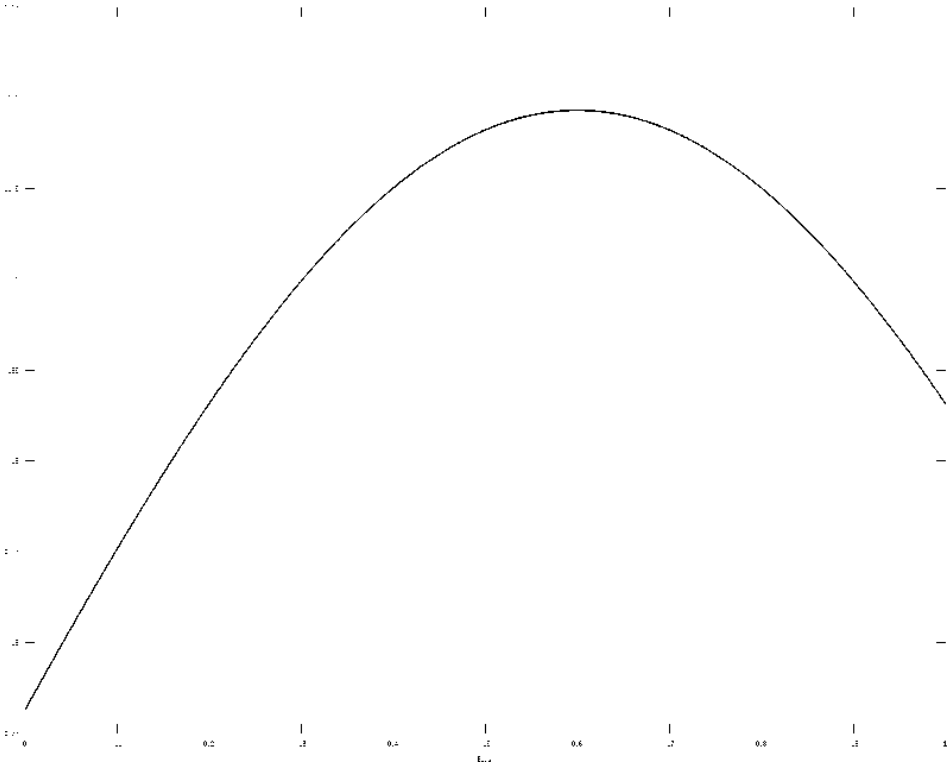
yielded a compression ratio of 92%, instead of the 88% obtained with 20 devices. Since the cells are independently drawn, there is no correlation between them, and thus the compression ratio obtained must be from the skewness in the bias distribution. This indicates that one has to be careful to say that CTW compression is a sign of correlation between PUFs, as compression can also result from biasing of the cells. The entropy estimation result for the CTW compression test, 0.88, is almost equal to the theoretical maximal entropy with two bins as denoted in Table 7.2. This suggests that CTW compression test is a good estimator for creation entropy in the scenario that only one enrollment measurement is performed (although more tests are required to confirm this).

The results of the entropy per cell as obtained from the Daugman inspired method (shown in Table 7.4, are remarkably high. The degrees of freedom, N, is very large compared to the number of cells used (8192). This effect is also visible in the plot, where the fit of distances obtained from synthetic data is much narrower than that of real PUF data. This indicates that the variance of the distances is much smaller, resulting in more degrees of freedom. This can probably be attributed to the fact that in our synthetic experiment, all PUF cell biases are drawn independently, thus no correlation between cells is introduced. In real PUF data, there may be correlations between cells. Furthermore, there is also some inaccuracy of the fit.

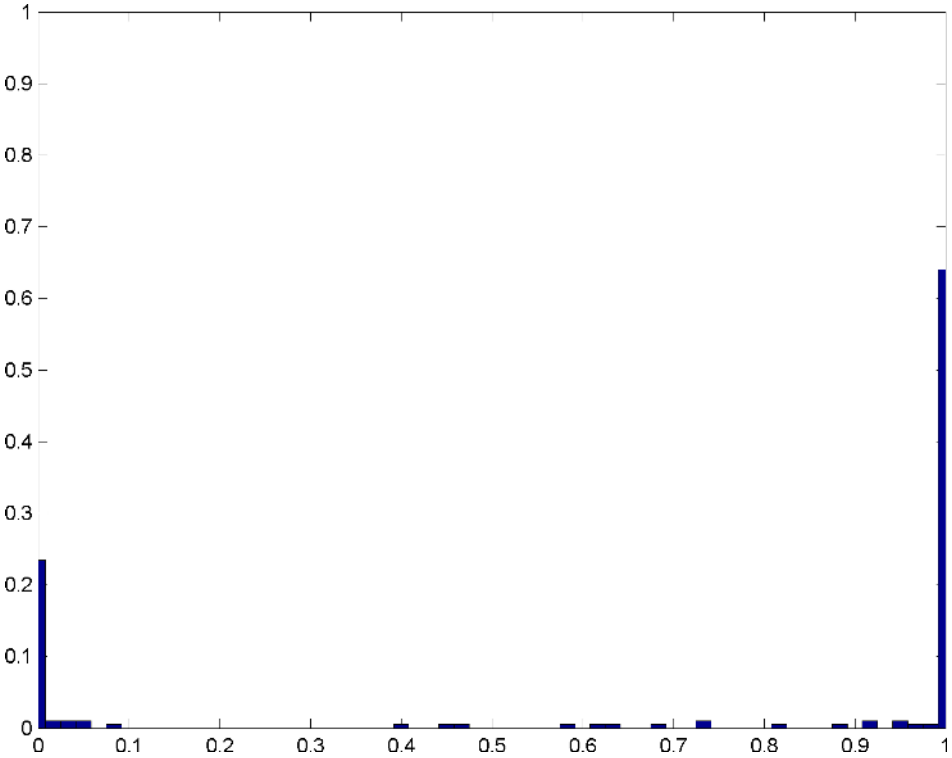
In contrast to the CTW compression test and min-entropy estimation method, the results of the Daugman inspired method do change when using more PUF devices during the analysis. When the number of synthetic devices is increased from 20 to 192, the estimated creation entropy decreases. This is caused by a decrease in the degrees of freedom N, which is shown in Table 7.4. In addition, with increasing number of sample devices the estimated average bias distance p decreases, but the influence of this effect on the estimated entropy is small. A remark has to be placed on the accuracy of results obtained using only 20 sample PUFs, as repetitive executions of this method yielded very different results, including results with lower entropy estimations than when using 192 PUFs. Therefore, this method is only recommended with a large enough sample size. Furthermore, by increasing the number of sample measurements (k), the same effect as when reducing the number of sample PUFs is observed: the histogram moves slightly to the left — indicating a minor decrease in bias distances between PUFs — and the histogram becomes much narrower. This again results in more degrees of freedom, and thus a higher creation entropy estimation. This effect is graphically depicted in Figure 7.5.

Lastly, the mutual information between synthetic enrollment and reconstruction data was calculated to obtain an estimate of the extractable entropy. The approximation of the actual extractable entropy becomes quite close to the estimated optimal extractable entropy of 0.46 when the number of measurements during enrollment and reconstruction increases. With the maximal number of enrollment and reconstruction measurements, the difference between the theoretical maximal extractable entropy bit per cell) and the estimation is only 0.01 (consider Table 7.5).

A result worth mentioning is that the extractable entropy obtained with a limited number of enrollment and reconstruction measurements lies below the min-entropy estimation. This is not strange, as the reconstruction conditions are omitted from the min-entropy measure, but does mean that if one counts on the min-entropy to obtain a lower bound on the entropy, this entropy might not be achieved in practice. For example, if one assumes that the number of distinguishable PUFs is equal to $2^{H_\infty(X)}$, these PUFs can all be uniquely identified under enrollment conditions. However, it might occur that two distinct PUFs are mistakenly identified as the same PUF when used under reconstruction conditions.



(a) Probability density function of the bias of a DFF cell as obtained from the maximal entropy method using approximated parameters λ_1 and λ_2.



(b) Probability mass function of the bias of a DFF cell as obtained from empirical estimation.

Figure 7.1: Estimated bias probability distributions.

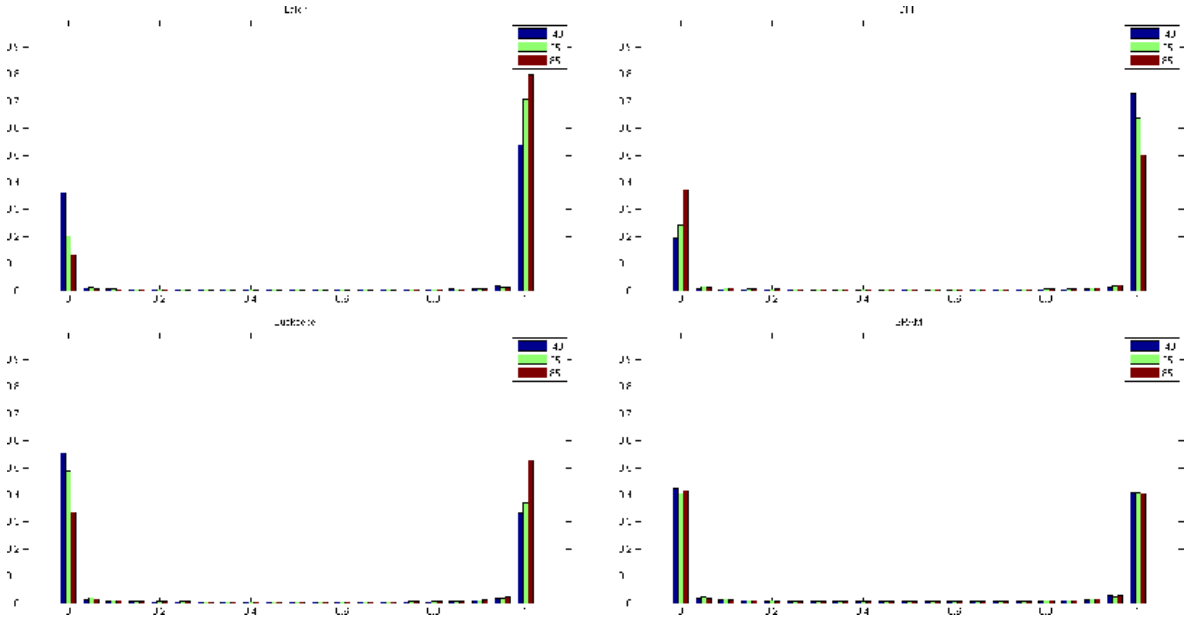


Figure 7.2: Histograms of the bias distribution of certain PUF Types (averaged over all cells). The PUF types associated with the histograms are the latch, DFF, buskeeper and SRAM PUF (from left to right, top to bottom). The different coloured bars denote the temperature under which the measurements were taken.

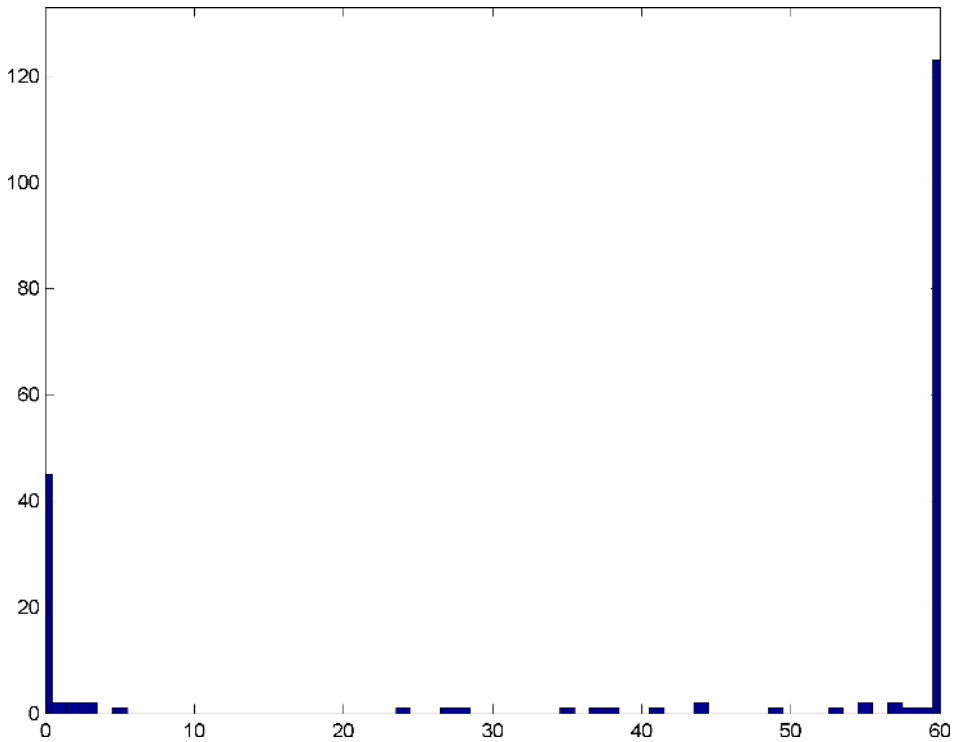
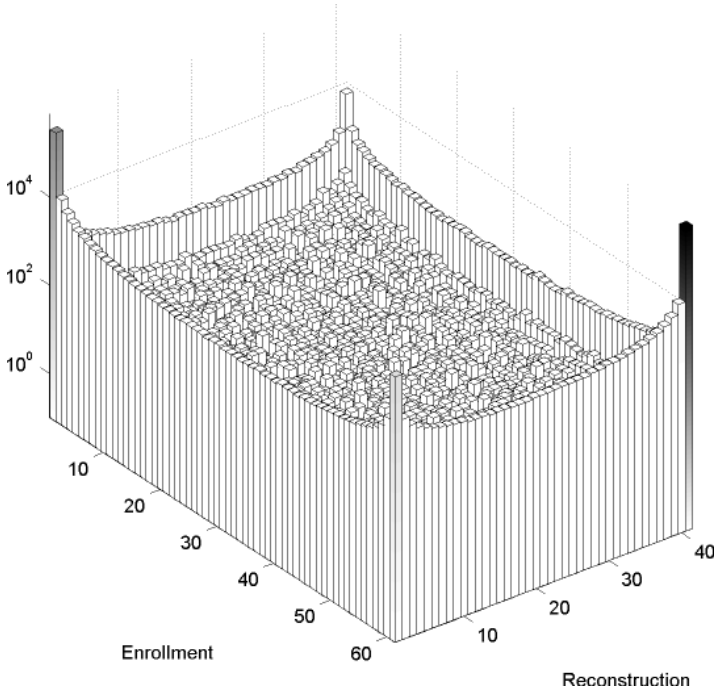
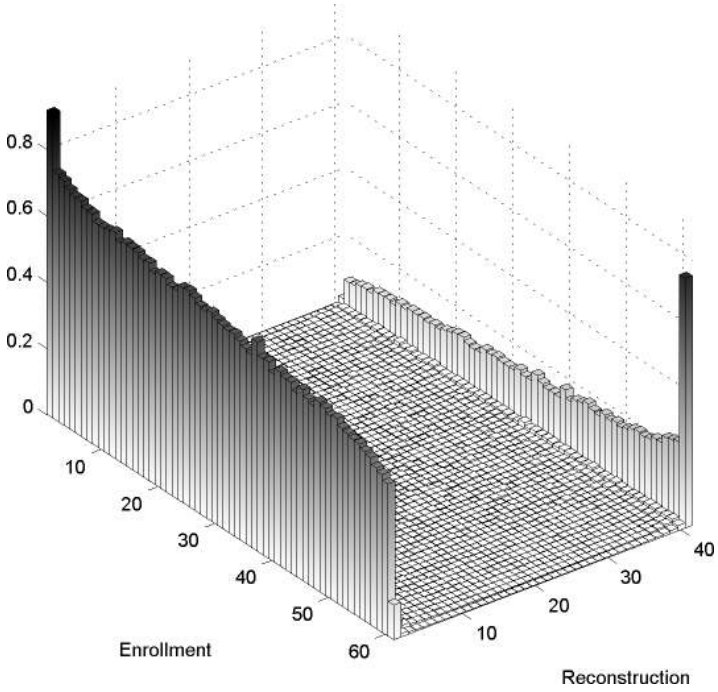


Figure 7.3: Histogram denoting the count (y-axis) of the number of times this cell started in the one state (x-axis).



(a) Observed changes in bias bins from enrollment to reconstruction measurements



(b) Plot denoting the global noise model. The bars indicate the probability of a cell having a certain bias during reconstruction given that the bias was in the bin denoted on the y-axis during enrollment: $p_{b_r|b_e}$.

Figure 7.4

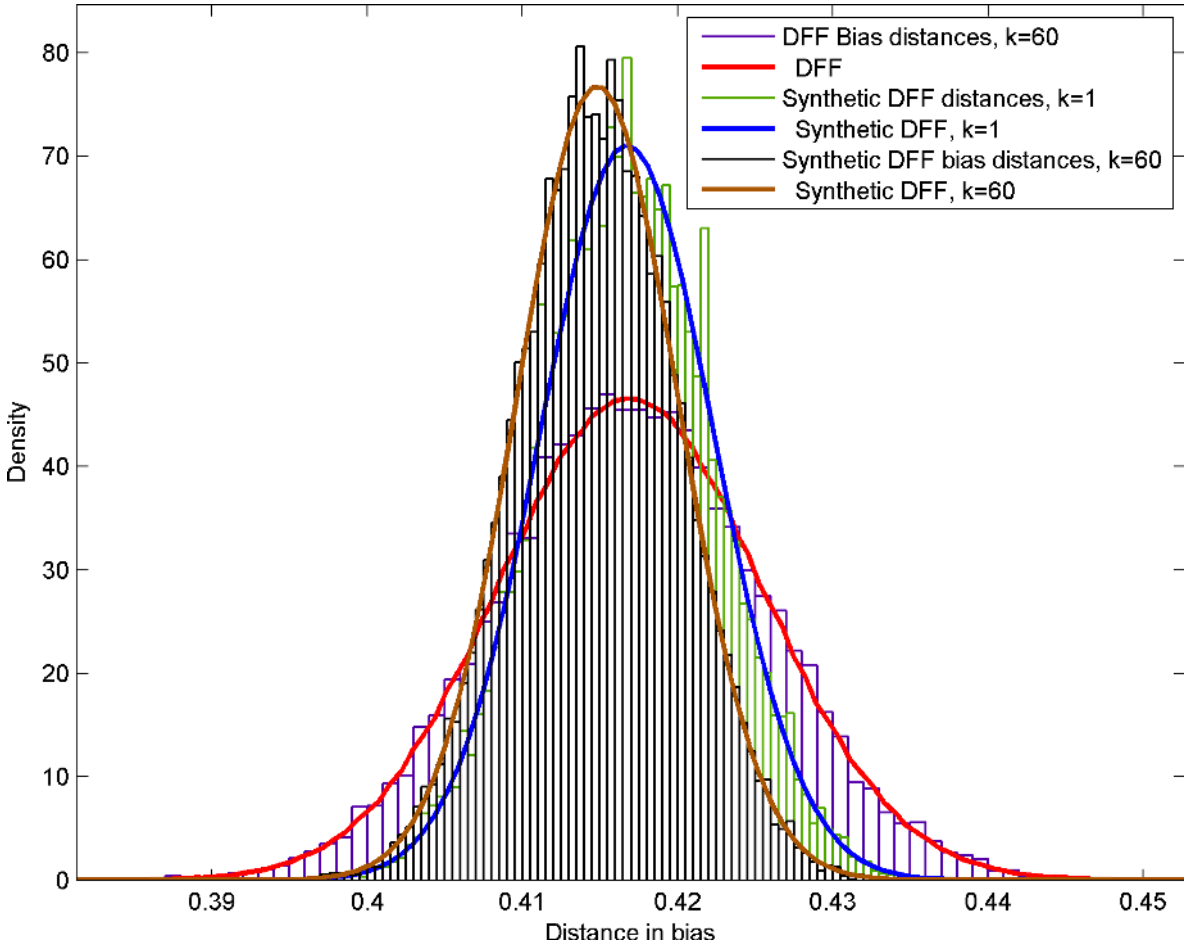


Figure 7.5: Histogram showing the distribution of bias distances (x-axis) between synthetic PUFs together with the distribution of bias distances of actual DFF PUFs. The histogram belonging to the actual DFF PUFs is clearly less narrow.

Chapter 8

Analysis of the UNIQUE PUF data

The goal of this chapter is to analyse the uniqueness and to estimate the entropy of PUF data using the methods described in the previous chapters. First, the PUF dataset obtained from the UNIQUE project is described. Then for each PUF type the analysis is performed and results are compared to those found in the literature. Lastly, the conclusion stating which PUF type contains the most entropy is reached.

8.1 Description of the data set

The UNIQUE project yielded 192 ASICs featuring the six PUF types described in Chapter 3.7. Here, the four memory based PUF types will be analysed. Each ASIC has four instantiations of the latch, DFF and SRAM PUF and two instantiation of the buskeeper PUF. Of each PUF type, one of the instances has been placed in a separate power domain. This can be used to cycle the power to these instances separately from the chip, which allows to obtain start-up values of these instances without restarting the entire chip. Furthermore, these instances can be powered off during normal usage to reduce ageing effects [25]. For this analysis, this functionality is not used.

Unfortunately two latch PUFs per ASIC are unusable due to faults in the addressing logic. Furthermore, during preliminary testing, one DFF instance was found to be very unreliable when compared to the other instances (also noticed by [25]). Hence, this instance is also excluded from the test data. This leaves a total of $2 \cdot 192 = 384$ latch and buskeeper PUFs, $3 \cdot 192 = 576$ DFF and $4 \cdot 192 = 768$ SRAM PUFs available for analysis. It is rare to have this quantity of PUFs available for analysis, most literature encountered performed entropy estimations on test sets with less than 20 PUFs.

All these PUF types provide 8192 bits of output, except the SRAM PUF which has 65536 bits of output. However, during the entropy analysis only 8192 out of these 65536 are used to reduce the required memory for processing. Depending on the complexity of the used entropy estimation method, even less cells are used to reduce the processing time and memory.

In the UNIQUE project, temperature and voltage variation tests were used to determine reliability. In this thesis, the data from this temperature variation test is used for the uniqueness analysis, since it provides both PUF responses obtained at room temperature and responses obtained at the standard operational temperature limits of electronics. The room temperature measurements are ideal candidates for enrollment measurements, while the measurements at other temperatures provide reconstruction conditions.

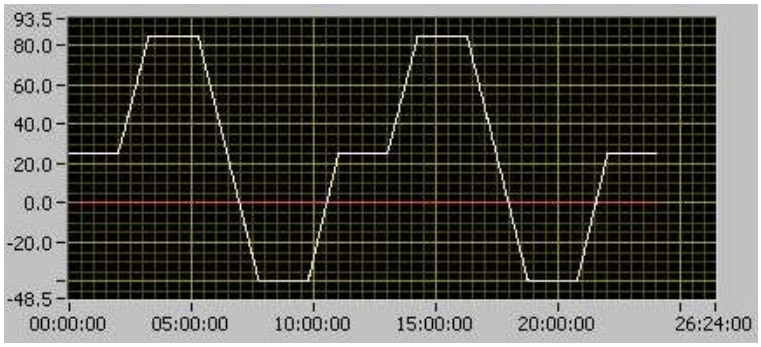


Figure 8.1: This figure illustrates the temperature profile of the climate chamber.

For the temperature variation test, the ASICs are placed in a climate chamber. First, 20 measurements are performed at 25 °C. Then the temperature is increased to 85 °C, and, once stable, 20 measurements are performed. After this, the temperature is decreased to -40 °C. Again, 20 measurements are performed once the temperature is constant. This procedure is repeated and concluded with 20 measurements at 25 °C. This temperature profile is displayed in Figure 8.1. The result is a total of 60 measurements at room temperature (used as enrollment measurements), and 40 measurements at -40 °C and 85 °C (used as reconstruction measurements). An example measurement showing the power-up values of 8192 D flip-flops (64 rows by 128 columns) is shown in Figure 8.2.

The actual analysis was performed on a 32bit 3GHz dual core PC with 2GB ram, using Matlab. To process the PUF data with Matlab, data matrices are created with cells as columns and measurements as rows. This is repeated for the number of PUFs, creating a `NrOfMeasurements` × `NrOfCells` × `NrOfPufs` 3D matrix per PUF instance.

The memory size required for these matrices can become rather large as the number of elements of these matrices increases. For example, if all cells of the SRAM PUF would be used, a $60 \times 65536 \times 192 = 754974720$ element matrix would be required to store enrollment data. However, as some Matlab functionality only works with (64-bit) doubles, these matrices cannot be stored and processed efficiently.

8.2 Entropy analysis of UNIQUE data

In this section, the entropy estimation methods are applied to the described dataset. First, for every method a quick recap is given followed by the results. Then, the results are compared to those found in the literature and finally an overview is given of the uniqueness performance of the various PUF types.

8.2.1 Reliability results

Before starting with the entropy analysis, a small assessment of the reliability across reconstruction conditions is performed. With the framework introduced in this thesis, PUF reliability became a rather vague term. Normally, a PUF is considered reliable when its response behaviour does not change over multiple start-ups (under possibly varying conditions): cells that start always as a one or always as a zero are reliable. Since the framework essentially considers a cells bias to be a (identifying) property — where biases not equal to zero or one are not a problem — bias

Figure 8.2: This figure illustrates a DFF PUF response reshaped to a 64 by 128 grid denoting the cell start-up states of a single measurement. White squares indicate the cell at that position powered-up as a '0', while a black square indicates a '1'.

robustness across reconstruction conditions is assessed instead. Thus, a cell bias of 0.6 is fine and as long as it stays the same under all conditions in which the PUF is used, it is robust.

To assess this robustness, for each PUF type an intra-device bias distance test is performed. The average intra-device distance among all devices can be calculated using the following equation:

$$\frac{1}{Pn}\sum_{p=1}^{P}\sum_{i=1}^{n}\left|\frac{x_i^p}{k}-\frac{y_i^p}{l}\right| \tag{8.1}$$

The results of this test is given in Table 8.1. The intra-device bias distance test shows that the robustness of a cells bias strongly depends on the PUF type and the reconstruction condition. For example, the bias of a latch PUF changes on average with more than 20% when reconstruction occurs at $-40°$C. Furthermore, the robustness of the latch PUF is higher when reconstruction occurs at high temperatures, while the DFF and buskeeper PUFs perform better at low temperatures. The SRAM PUF is in both conditions very robust.

8.2.2 Correlation

Pearson's product-moment coefficient is calculated for every PUF instance to determine if there is any correlation among the cells. Although zero correlation does not directly imply independence, any correlation found during testing would indicate that there exists linear dependencies between

Instance	Average intra-device distance		Maximum intra-device distance	
	−40°C	+85°C	−40°C	+85°C
Latch 1	0.231	0.103	0.274	0.171
Latch 2	0.233	0.117	0.277	0.182
DFF 1	0.125	0.176	0.158	0.194
DFF 2*	0.153	0.174	0.318	0.217
DFF 3	0.122	0.178	0.157	0.196
DFF 4	0.120	0.177	0.166	0.197
Buskeeper 1	0.090	0.172	0.099	0.196
Buskeeper 2	0.092	0.171	0.101	0.200
SRAM 1	0.054	0.050	0.059	0.056
SRAM 2	0.053	0.050	0.060	0.057
SRAM 3	0.053	0.050	0.059	0.057
SRAM 4	0.053	0.050	0.061	0.058

Table 8.1: Table denoting the bias robustness results of UNIQUE PUFs. The values denote how much the bias changes on average due to temperature influences in comparison to the bias under the enrollment condition. The DFF PUF instance 2 is excluded from the entropy analysis due to being unreliable.

cells. In the literature, it is assumed that PUF cells are independent [41, 8].

For this test, the first 1024 cells are used. Pairwise, the covariance of two cells is divided by the product of their standard deviations as shown in Equation 8.2. The result is a value between -1 and 1, where 1 denotes a very strong positive relation and -1 denotes a very strong negative relation, which means that when the bias of cell i increases, the bias of cell j decreases. The closer this value lies to zero the weaker the relationship between the two cells.

$$\mathrm{Corr}(x_i, x_j) = \frac{\mathrm{Cov}(x_i, x_j)}{\sigma_{x_i}\sigma_{x_j}} \tag{8.2}$$

Furthermore, we calculated the probabilities of getting a correlation as large as observed under the hypothesis that there is no correlation. When this probability is less than 0.01, a correlation is considered significant.

For all instances, the percentage of cells failing the hypotheses of no correlation lies around 0.010 with a maximum of 0.014 for the latch PUF. This amount of significant correlations is exactly what can be expected by chance. Furthermore, from the significant correlations, the strength of the correlation is approximately 0.2. When the correlation test is applied to synthetically generated PUF data (known to be independent), similar values are observed. These results indicate that *linear* correlation among cells is very low or non-existent.

8.2.3 Weights and distances

Using the matrices with PUF data, it is trivial to analyse possible biasing of PUF types using the Hamming weight measure. If we sum along the rows (measurements), a value between 0–60 is obtained denoting how often out of the 60 enrollment measurements this cells' power-up value was one. By dividing by 60, the fractional Hamming weight per cell is obtained which can be used to determine positional biasing. For all PUF types[1], Figure 8.3 shows the fractional Hamming weight per cell under enrollment conditions, where the cells are displayed in a 64×128 grid.

[1]Only the results of the first instance is displayed (192 devices), but the results among the instances are the same.

Latch	DFF	SRAM	Buskeeper
0.75	0.70	0.50	0.44

Table 8.2: Average fractional Hamming weight per PUF type under enrollment conditions.

From this figure, it is clear that only the SRAM PUF suffers from a little bit of positional biasing, that is, a correlation between cell position and bias is present. The figure shows that the cells of rows 1–16 and 33–48 have biases closer to 0.55 while the other rows have a lower bias around 0.45. This might be caused by the fact that SRAM memory consists of 2 banks.

Furthermore, the amount of variation in biases between cells is different for the four PUF types. The DFF PUFs biases seem very similar over all cells, as in the bar graph the surface is almost flat. The buskeeper PUF has strongly varying biases as its graph has a lot of peaks and dips. The latch PUF is somewhere in between with less variation than the buskeeper PUF but more than the DFF PUF.

The figure also illustrates that some PUFs suffer from systematic biasing. For example, the latch and DFF PUFs generally have a high bias. This indicates that the cells of these PUF types generally favour the 'one' start-up state. This is reinforced by Table 8.2 that shows the average fractional Hamming weight of all PUF types, sometimes also denoted as uniformity [32]. This total average fractional Hamming weight is determined by calculating for each PUF type for all instances, measurements and cells the fraction of start-ups that resulted in a '1' out of all start-ups. Moreover, there are some cells that have a very high systematic bias and hence contain almost no information.

This table shows that the buskeeper PUF is also slightly biased (but towards zero) and the SRAM PUF has an average bias close to 0.5, which is optimal. It can be concluded that all PUF types analysed, except the SRAM PUF, have some biasing, even without considering temperature changes.

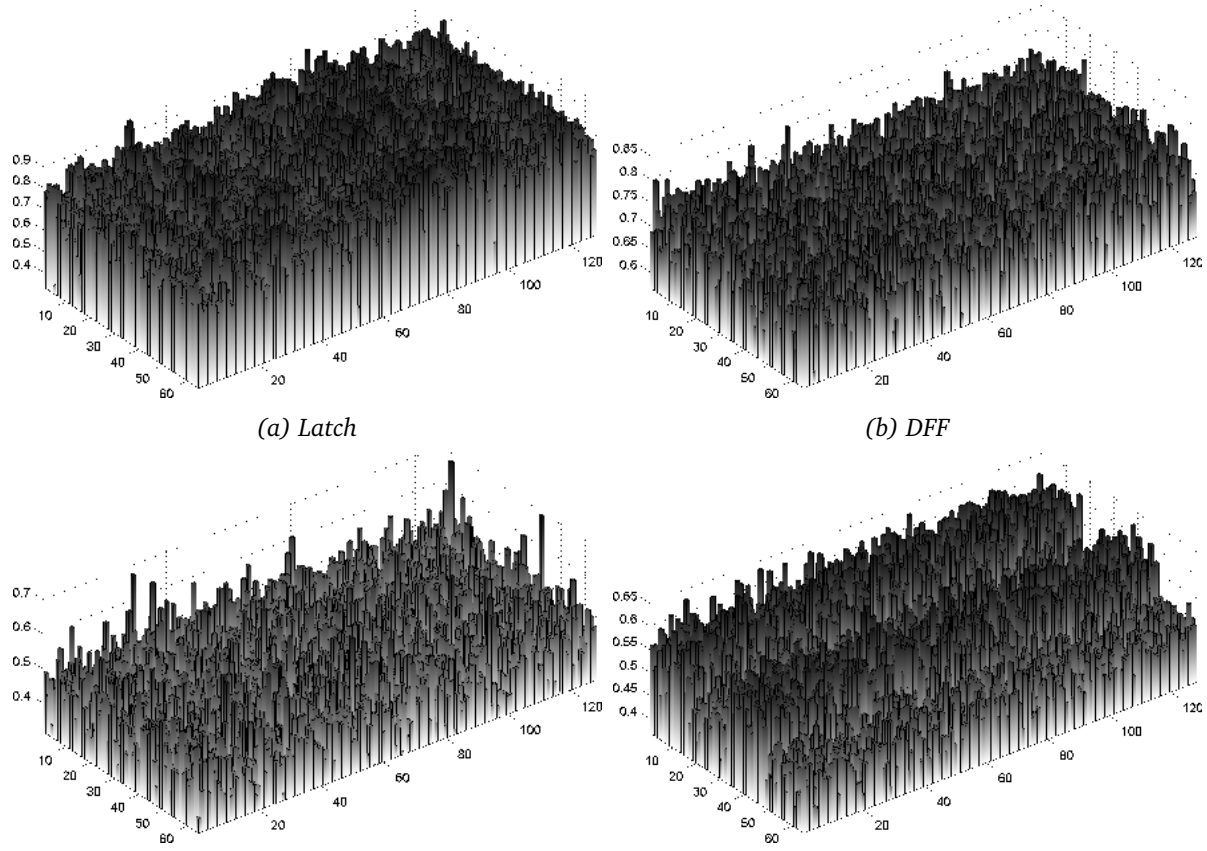
By summing along the rows and columns of the data matrix, and dividing by both the number of measurements and the number of cells, a fractional Hamming weight per PUF is obtained. The results are displayed in Figure 8.4. This figure shows that all PUFs behave somewhat similar, there are no outliers that have a totally different fractional Hamming weight. Again, the PUF type with the most variation between Hamming weights is the buskeeper PUF.

8.2.4 Inter-device distance

In the inter-device distance test, the difference between two PUFs of the same type is assessed. With our framework, the distance means the average difference in cell bias, averaged over all cells. This is different from the inter-device distance test handled in Section 4.1, as there the fractional Hamming distance is considered, thus only using one measurement per PUF. When using multiple measurements, the distance between PUF p and p' can be expressed as:

$$distance(p, p') = \frac{1}{nk} \sum_{i=1}^{n} |x_i^p - x_i^{p'}| \tag{8.3}$$

Note that if the number of measurements k is one, this is just the Hamming distance in which case it is equal to the measure previously discussed. Preferably, the distance between two PUFs is close to 0.5, which means that correlation among PUFs is low. This is an indication of uniqueness, as knowing one PUFs biases, one has maximal uncertainty about other PUFs biases.



(a) Latch *(b) DFF*

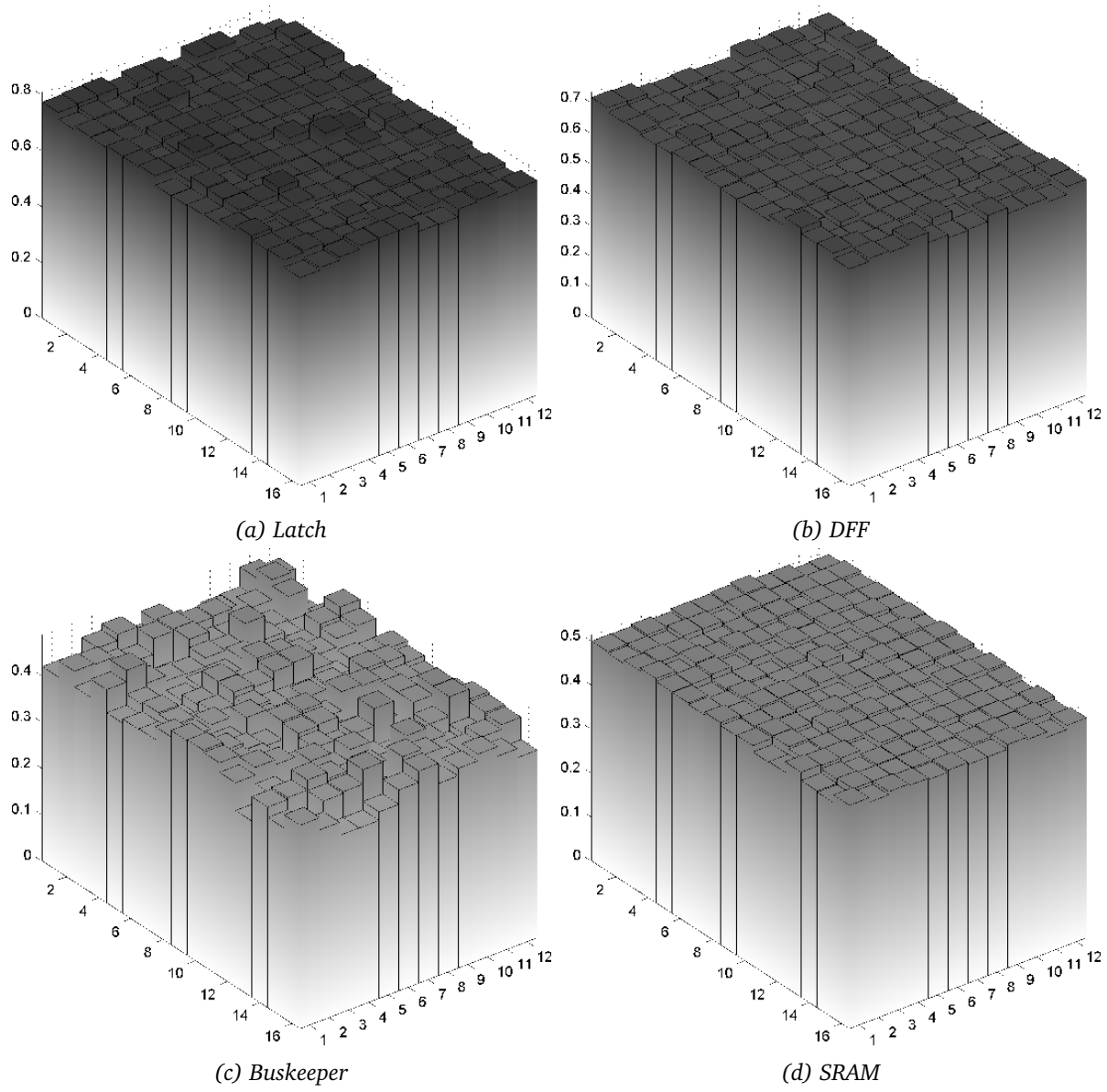
(c) Buskeeper *(d) SRAM*

Figure 8.3: Estimated cell bias per PUF type, where the height denotes the bias and the cells are represented in a 2D grid of 64 by 128 cells. Note that the scale of the z-axis (height) can differ per PUF type.

Here, the inter-device distance is considered across reconstruction conditions. This is not often done, most analyses consider reliability and uniqueness separately in respectively an intra-device distance test across reconstruction conditions and an inter-device distance test on enrollment data. Since for the extractable information, uniqueness and reliability are intertwined, we think it provides a good indication for how much information can be extracted in practice.

Therefore, a pairwise comparison of all 192 devices (per PUF type the first instance is used) under all 3 temperatures is performed. This gives a total of $\frac{576*575}{2} = 165600$ comparisons. These comparisons are graphically represented in a 3D-bar graph of Figure 8.5. In this graph, both the horizontal as vertical axis contain devices numbers, where the values denotes the average difference in bias as given in Equation 8.3. The device number denotes 3 times the same 192 devices, although the first 192 resulted from reconstruction measurements performed at $-40°$C, the second 192 from enrollment measurements at room temperature and the last 192 from reconstruction measurements at $+85°$C. In this way, 9 'tiles' are created, where for example the second one on the top row indicates the comparison of devices measured at -40 (rows) with those measured at 25°C (column). Note that the figure is symmetric, the comparisons are displayed twice.

The same information, but displayed differently, is given in Figure 8.6. This figure shows more detail, but depends on colour information. The main diagonal contains all zeros, because



(a) Latch *(b) DFF*

(c) Buskeeper *(d) SRAM*

Figure 8.4: Estimated average PUF bias of 192 devices per type. The PUFs are arranged in a grid of 12 by 16 devices. The height denotes the average bias over 8192 cells.

this compares the same device with itself under the same condition. The other diagonal lines are the result of comparing a device with itself under different conditions. This information is lost in the bar graph, but that graph shows better that there is also variation when comparing devices under the same condition, e.g. the spikes in the squares.

These figures both show that inter-device distances strongly varies between devices and conditions. For example, the DFF PUF actually performs better when used at high temperatures. This is clearly visible in the last 'tile', where the inter-device distance is close to 0.5. On the other hand, if the PUF was enrolled at room temperature and used at -40°C, the average inter-device distance is only 0.37 in comparison with 0.42 when used at room temperature. This hints that extractable information probably decreases when the PUF is used at lower temperatures.

The variation between distances under the various conditions is the largest with the latch PUF, followed by the DFF and then the buskeeper PUF. The inter-device bias distances between SRAM PUFs are unaffected by temperature conditions and seem to be very optimal as they lie around 0.5.

8.2.5 Daugman inspired method

By considering the inter-device bias distances under the enrollment condition (the central 'tile' in the inter-device distance figures), the adapted Daugman method can be used to obtain an entropy estimation.

For several values of k, denoting the number of enrollment measurements used, these inter-device distances are calculated. Since we consider the difference between distinct PUFs as binomially distributed with parameters N and p, a binomial fit can be applied to the histogram of these distances. For $k = 60$, the maximum number of enrollment measurements, this fit is displayed in Figure 8.7. In these figures, also the histograms of the intra-device bias distances for the two reconstruction conditions is given.

This figure clearly shows that all PUF types present on the UNIQUE chip exhibit PUF behaviour, i.e. the difference between devices is always larger than within devices, but that some PUF types are in this respect more suitable than others.

In Figure 8.8, the binomial fit of the inter-device distances of the four PUF types are depicted in the same plot. This is shown for $k = 1$ (comparing Hamming distances), and for $k = 60$ (using the maximum number of measurements available, thus comparing bias distances with the largest number of distinguishable biases). This figure shows that the distribution of distances of the SRAM and buskeeper are much narrower and lie closer to 0.5 than that of the latch and DFF PUF. Especially the distribution of the latch PUF is very wide and its center lies far from the optimal bias distance 0.5, indicating a low number of trials N and low probability p as parameters. Using Equation 6.21, the entropy of each instance is estimated. For various number of enrollment measurements k, the resulting entropy estimation is shown in Table 8.3.

Instance	Entropy per cell							
	k=1	k=2	k=3	k=5	k=10	k=20	k=30	k=60
Latch 1	0.19	0.18	0.18	0.19	0.19	0.19	0.19	0.19
Latch 2	0.19	0.19	0.19	0.18	0.19	0.19	0.19	0.18
DFF 1	0.42	0.46	0.46	0.47	0.47	0.47	0.47	0.46
DFF 3	0.37	0.39	0.39	0.40	0.39	0.40	0.40	0.40
DFF 4	0.39	0.40	0.40	0.40	0.41	0.41	0.41	0.42
Buskeeper 1	0.61	0.65	0.67	0.68	0.69	0.69	0.67	0.67
Buskeeper 2	0.67	0.70	0.71	0.72	0.73	0.73	0.73	0.72
SRAM 1	0.81	0.87	0.91	0.91	0.95	0.95	0.96	0.96
SRAM 2	0.82	0.89	0.91	0.94	0.96	0.96	0.98	0.98
SRAM 3	0.83	0.90	0.92	0.95	0.97	0.97	0.98	1.00
SRAM 4	0.85	0.93	0.97	0.98	1.00	1.03	1.02	1.04

Table 8.3: For different number of measurements used k, this tables shows the resulting entropy using the modified Daugman method.

A PUF type that has a low inter-device bias distance, such as the latch PUF, does not benefit from an increasing number of measurements at all. In fact, a small decrease in entropy seen as

the number of measurements k increases which can be attributed to inaccuracies in the fit or numeric noise. A finer distinguishable bias starts to make a difference when the inter-device bias distance is close to 0.5, as is the case with the Buskeeper and SRAM PUF. Especially the last type sees an increase in entropy of around 20%.

From the fitted probability density function a binomial cumulative is plotted in Figure 8.9. For the bias distance ranging from 0.2–0.5, the cumulative probability is also displayed in Table 8.4. This table provides a measure of uniqueness in the sense that, if two PUFs were compared as a means of identification, and a distance in bias between the PUFs cells was found to be 25%, the probability that these responses originate from two different PUFs is negligible for all PUF types present on the UNIQUE ASIC (provided that the measurements occurred at enrollment conditions). The graphical CDF shows that the distance at which this probability becomes non negligible differs for the various PUF types, e.g. the probability of false identification is much lower for the SRAM PUF than for the latch PUF under the same bias distance.

	$\Pr(distance < X)$							
	Latch		DFF		Buskeeper		SRAM	
X	k=1	k=60	k=1	k=60	k=1	k=60	k=1	k=60
0.20	1.4^{-041}	1.1^{-039}	2.8^{-173}	7.6^{-187}	0	0	0	0
0.21	4.6^{-036}	2.5^{-034}	2.4^{-157}	3.1^{-169}	0	0	0	0
0.22	2.7^{-031}	1.1^{-029}	2.5^{-142}	4.5^{-153}	0	0	0	0
0.23	1.2^{-026}	3.5^{-025}	3.4^{-128}	1.8^{-137}	1.3^{-313}	0	0	0
0.24	1.1^{-022}	2.4^{-021}	6.4^{-115}	3.4^{-123}	1.9^{-288}	1.1^{-308}	0	0
0.25	4.5^{-019}	1.2^{-017}	1.8^{-102}	1.8^{-109}	6.2^{-265}	4^{-283}	0	0
0.26	1.2^{-015}	1.5^{-014}	8^{-091}	5.6^{-097}	1.4^{-242}	8^{-259}	0	0
0.27	9.1^{-013}	1.2^{-011}	5.7^{-080}	4.8^{-085}	6.1^{-221}	9.2^{-236}	8^{-310}	0
0.28	3.1^{-010}	3^{-009}	1.3^{-069}	3^{-074}	7.9^{-201}	6.6^{-214}	1.7^{-282}	6.3^{-321}
0.29	6.7^{-008}	4.6^{-007}	2.8^{-060}	5.3^{-064}	8.8^{-182}	3.1^{-193}	3.5^{-256}	1.6^{-290}
0.30	4.9^{-006}	2.5^{-005}	1.1^{-051}	8^{-055}	8.7^{-164}	2.2^{-173}	2.8^{-231}	8.7^{-262}
0.31	0.00018	0.0008	7.7^{-044}	3.3^{-046}	1.7^{-146}	5^{-155}	4^{-208}	5.1^{-235}
0.32	0.0037	0.01	1.1^{-036}	1.4^{-038}	1.5^{-130}	8.6^{-138}	5.6^{-186}	1.7^{-209}
0.33	0.033	0.077	3.3^{-030}	1.6^{-031}	1.3^{-115}	1.2^{-121}	3.6^{-165}	1.5^{-185}
0.34	0.17	0.28	2^{-024}	2.2^{-025}	2.3^{-101}	1.4^{-106}	5.9^{-146}	3.7^{-163}
0.35	0.45	0.59	2.6^{-019}	5.6^{-020}	2.3^{-088}	1.4^{-092}	9.5^{-128}	2.7^{-142}
0.36	0.76	0.87	7.4^{-015}	3.7^{-015}	2.6^{-076}	1.3^{-079}	7.9^{-111}	6.1^{-123}
0.37	0.94	0.97	4.8^{-011}	3.6^{-011}	3.4^{-065}	1.1^{-067}	2.1^{-095}	4.5^{-105}
0.38	0.99	1	7.1^{-008}	8.5^{-008}	8.1^{-055}	9^{-057}	5.7^{-081}	1.1^{-088}
0.39	1	1	2.9^{-005}	3.5^{-005}	1.5^{-045}	7.1^{-047}	8.7^{-068}	1^{-073}
0.40	1	1	0.0024	0.0035	3.3^{-037}	5.6^{-038}	5.4^{-056}	3.2^{-060}
0.41	1	1	0.053	0.074	9.6^{-030}	4.6^{-030}	3.1^{-045}	3.9^{-048}
0.42	1	1	0.34	0.43	4.8^{-023}	3.9^{-023}	1.1^{-035}	1.8^{-037}
0.43	1	1	0.78	0.86	2.3^{-017}	3.6^{-017}	2^{-027}	3.4^{-028}
0.44	1	1	0.98	0.99	1.5^{-012}	3.6^{-012}	3.2^{-020}	2.5^{-020}
0.45	1	1	1	1	1.5^{-008}	4^{-008}	3.5^{-014}	8^{-014}
0.46	1	1	1	1	1.9^{-005}	5.2^{-005}	2.2^{-009}	1.1^{-008}
0.47	1	1	1	1	0.0034	0.0083	1.2^{-005}	6.8^{-005}
0.48	1	1	1	1	0.097	0.18	0.0049	0.021
0.49	1	1	1	1	0.56	0.72	0.17	0.4
0.50	1	1	1	1	0.94	0.98	0.75	0.94

Table 8.4: Cumulative probabilities of inter-device bias distances

8.2.6 Min-entropy

For every PUF instance, the bias probability distribution of each cell is estimated and the maximal probability is determined. This is done using sixty measurements per PUF and only one measurement per PUF. The maximum probability is then fed to Equation 2.4 to calculate the min-entropy.

The final min-entropy per cell is obtained by averaging the min-entropy over all cells, which is displayed in Table 8.5.

For the single measurement scenario, the latch PUF performs worst with a min-entropy of 0.39 bit per cell. The DFF PUF is slightly better with on average half a bit of entropy per cell. The buskeeper PUF is clearly better with a min-entropy of 0.82 and the SRAM PUF has the largest min-entropy with 0.87. An overview of the results of all instances is given in Table 8.5. When using 60 measurements, the resulting min-entropy per cell is higher but the ranking of the PUF types stays the same.

These results are not surprising as Table 8.2 denoting the average fractional Hamming weight per PUF already 'predicts' the expected min-entropy, since PUFs that are biased on average will have cells that are biased, which in turn lowers the min-entropy score.

Instance	Min-entropy (k=1)	Min-entropy (k=60)
Latch 1	0.39	0.50
Latch 2	0.43	0.55
DFF 1	0.53	0.70
DFF 3	0.51	0.68
DFF 4	0.50	0.66
Buskeeper 1	0.82	1.06
Buskeeper 2	0.82	1.06
SRAM 1	0.87	1.21
SRAM 2	0.88	1.21
SRAM 3	0.88	1.22
SRAM 4	0.88	1.22

Table 8.5: This tables shows the min-entropy estimation results using 192 devices per instance.

8.2.7 CTW-Compression

For each PUF instance, the CTW compression test is performed. To reduce the required time for this test, only 96 out of the 192 devices per instance are used.

For each device, the first enrollment measurement is taken and concatenated in a single bit string. By running the CTW compression test on this bit string, a compression ratio is obtained. A compression ratio lower than 1.00 indicates non-randomness and hence a lack of entropy. The results of this test are given in Table 8.6.

This table illustrates that the SRAM PUF performs very well as the concatenated responses from this type of PUF are not compressible. The buskeeper PUF is a close second with only 1% compression. The latch and DFF PUF perform not so well, indicating that they might not have full entropy.

In addition, for each PUF instance a bit string containing concatenated enrollment responses of the same devices is compressed to determine the intra-device compression ratio. Compression occurs because of the similar responses, but less biasing in this bit string leads to less compression, hence the varying results in Table 8.6.

Instance	Inter-device compression ratio	Intra-device compression ratio
Latch 1	0.83	0.13
Latch 2	0.81	0.14
DFF 1	0.88	0.16
DFF 3	0.87	0.15
DFF 4	0.86	0.15
Buskeeper 1	0.99	0.20
Buskeeper 2	0.99	0.18
SRAM 1	1.00	0.22
SRAM 2	1.00	0.24
SRAM 3	1.00	0.22
SRAM 4	1.00	0.22

Table 8.6: This tables shows the CTW compression results per PUF instance. The first column denotes the compression ratio obtained by concatenating the first enrollment responses of 96 different devices. The second column is the compression ratio obtained by concatenating 60 PUF enrollment responses of the same device, averaged over 96 PUF devices. It shows that PUF types that have (roughly) an equal amount of ones and zeroes in their responses (unbiased) are harder to compress.

8.2.8 Mutual Information

For various values of k and l, the mutual information of the four different PUF types is shown graphically in Figure 8.10. On the x and y-axis, the number of measurements used during enrollment and reconstruction is displayed while the height denotes the mutual information using that configuration. The general trend is that with increasing number of enrollment and reconstruction measurements, the mutual information rises. However, the extent with which the mutual information rises is strongly dependent on the PUF type. Especially PUFs that are less effected by environmental conditions, such as the buskeeper and SRAM PUF, benefit from using multiple measurement.

In addition, for three interesting scenarios, the mutual information is given in Table 8.7. In the first scenario, only one enrollment and reconstruction measurement is used. In the second, 60 enrollment measurements are used and only one reconstruction measurement. In the last scenario 60 enrollment measurements and 40 reconstruction measurements are used.

The conditions giving the lowest mutual information are marked with a star in Table 8.7. In this way, a lower bound on the extractable entropy per PUF type is obtained. Furthermore, in the section introducing PUF types present on the UNIQUE ASIC, Table 3.2 was given detailing the number of output bits per PUF type per square millimetre. Here, that table is revisited now considering the actual extractable bits using the worst reconstruction condition, i.e. providing a lower bound. The total extractable entropy is calculated for the three described scenarios by considering the worst performing instance per PUF type, and multiplying the average entropy per bit by the number of bits produced per square millimetre. The results are shown in Table 8.8, from which it is clear that the SRAM PUF has by far the most extractable bits followed by the buskeeper PUF.

Condition	Instance	Mutual information (per cell)		
		k=1,l=1	k=60,l=1	k=60,l=40
	Latch 1*	0.18	0.21	0.24
	Latch 2	0.20	0.23	0.26
	DFF 1	0.33	0.39	0.46
	DFF 3	0.33	0.39	0.45
	DFF 4*	0.33	0.38	0.44
−40 °C	Buskeeper 1	0.53	0.63	0.77
	Buskeeper 2	0.52	0.62	0.75
	SRAM 1*	0.61	0.77	1.08
	SRAM 2	0.61	0.77	1.08
	SRAM 3	0.61	0.77	1.08
	SRAM 4	0.62	0.77	1.08
	Latch 1	0.29	0.33	0.40
	Latch 2	0.28	0.32	0.39
	DFF 1	0.33	0.40	0.46
	DFF 3	0.32	0.39	0.46
	DFF 4	0.33	0.40	0.46
+85 °C	Buskeeper 1	0.43	0.53	0.66
	Buskeeper 2*	0.42	0.52	0.65
	SRAM 1	0.62	0.77	1.12
	SRAM 2	0.62	0.77	1.12
	SRAM 3	0.62	0.77	1.12
	SRAM 4	0.62	0.77	1.12

Table 8.7: This table denotes for each instances and each reconstruction condition the mutual information for three different combinations of k and l. The instance and reconstruction condition giving the lowest mutual information is marked with an asterisk.

8.3 Discussion of results

In this section, the results obtained in the analysis are compared to those found in other literature. After that, the most important results of the entropy analysis are recapitulated and the performance among PUF types is compared. A section summarising the effects that influence the extractable entropy concludes this chapter.

8.3.1 Comparisons with other work

The results obtained in this chapter are compared to the uniqueness estimation and entropy estimation results found in the literature, given in Table 4.1. First, note that that some of the entries in that table are obtained using the same dataset as is used in this thesis which is the PUF data from the UNIQUE project. In this respect, it is not surprising that our results regarding intra-device and inter-device distances are very similar to those obtained in [25], even though that work considered Hamming distances where in this thesis the difference between biases was used. This is a consequence of the bias probability distribution which has almost all probability concentrated near the edges, which means most cells have a bias of 0 or 1.

For some PUFs, such as the SRAM PUF, the result of the inter-device distance test seldom

PUF Type	Area (mm^2)	Bits per mm^2	Minimum extractable bits		
			k=1,l=1	k=60,l=1	k=60,l=40
Latch	.272	≈ 120470	21684	25298	28912
D Flip-Flop	.392	≈ 83592	27585	31764	36780
Buskeeper	.076	≈ 215579	90543	112101	140126
SRAM	.213	≈ 1230723	750741	947656	1329180

Table 8.8: Overview of the extractable bits per square millimetre occupied on the UNIQUE chip, broken down to PUF type and depending on the number of measurements used during enrollment (k) and reconstruction (l).

differs from the optimal result of 0.5 [17, 36, 8, 25, 39]. The results of the inter-device tests of this thesis shows the same result, and even when comparing the differences between enrollment PUFs and reconstruction PUFs, the optimal distance of 0.5 is observed. For other PUFs, such as the DFF PUF, the inter-device distance varies more. In [48], the lowest distance of 0.36 has been found and in [8] the best results has been found of 0.5. Our analysis, comparing bias distances, shows an average distance of 0.42 under enrollment conditions. For the other PUFs types, the latch and buskeeper PUFs, no inter-device distance results have been found except [25] which found exactly the same distances as resulted from our analysis.

For the creation entropy estimation methods determining the min-entropy per bit and CTW ratio, previous results are available for the SRAM, DFF and buskeeper PUF types. The estimated min-entropy and CTW ratio for the buskeeper is identical to [41], which is not surprising as the analysis was again performed on the same dataset. For the DFF PUF type, the results of the min-entropy and CTW compression test performed by [8] are higher than found in our analysis, even though we considered multiple possible biases in our test which generally leads to a higher min-entropy. As all results found in that work are higher than in our analysis, it might be possible that the DFF PUF tested in that paper was a better implementation. In general, entropy estimations of the DFF PUF type lie further apart than with SRAM PUFs. For the SRAM holds that all results found in the literature find a lower min-entropy than our estimation but the results for the CTW compression test are identical, namely that SRAM responses cannot be compressed. An exception to this is the SRAM proposal by [17], who found a compression ratio of 0.76 but there the CTW test is used to estimate the mutual information using both enrollment and reconstruction responses. This value is high compared to the estimation of the mutual information in our analysis, which was 0.62 bit per cell (also using one enrollment and one reconstruction measurement).

8.3.2 Recapitulation and conclusion

The best performing PUF analysed in this thesis is clearly the SRAM PUF, as it scores best on all uniqueness measures and entropy estimation methods. The only weakness found in the behaviour of the SRAM PUF was the slight positional biasing of its cells, where blocks of 16 consecutive cells have alternating average biases of respectively 0.55 and 0.45, where ideally the bias of all cells would lie around 0.50. Due to the good bias distribution of the SRAM PUF, the creation entropy is very high. The estimated entropy using the min-entropy method is 0.87 bit per cell. Using the modified Daugman algorithm with 1 and 60 measurements, the estimated entropy is respectively ≈ 0.83 and 0.99 bits per cell. The CTW compression algorithm could not compress the responses, thereby confirming the high creation entropy. Furthermore, due to the stability

of the SRAM PUF over all reconstruction conditions, the extractable entropy is also high, and increases steadily if more measurements are used during enrollment and reconstruction. In the case that the maximum number of enrollment and reconstruction measurements is used, this is the only PUF type tested that obtains an extractable entropy of more than one bit per cell, 1.08 to be precise. This is almost double the entropy that can be extracted if only a single measurement was used during enrollment and reconstruction (0.62 bit per cell).

The buskeeper performed slightly less but still provided excellent results, with an estimated min-entropy of 0.82 bit per cell. The modified Daugman method found entropies varying from 0.61 to 0.72 bit per cell, again using respectively 1 and 60 measurements. The compression ratio found using the CTW compression test was only 1%, indicating a very small entropy loss. The extractable entropy as estimated by the mutual information was negatively influenced by the instability of the biases under the reconstruction conditions, as can be seen in the histogram of Figure 7.2. Although the gain in extractable entropy using multiple measurements is not as big as with the SRAM, the extractable entropy for the buskeeper still benefits from multiple measurements and is generally high compared to the latch and DFF PUF, averaging between 0.42 and 0.75 bit per cell.

The results for the DFF and Latch PUF were less convincing. Especially the latter performed bad on some tests, predominantly due to a severely skewed bias distribution. The average estimated bias of the latch PUF is 0.75, indicating that on average 3 out of 4 cells will start in the one state. This directly influences the min-entropy which is estimated to be 0.39 bit per cell for the latch PUF, which is the lowest of the four analysed PUFs. The compression ratio obtained through the CTW test is 0.81 for the latch PUF, indicating a definite lack of entropy. Further entropy estimation using the Daugman method found an entropy of 0.19 bit per cell which does not increase using multiple measurements. Furthermore, the latch PUF has an extractable entropy between 0.18 and 0.40 bit per cell, mainly dependent on the reconstruction condition. The number of measurements is of little influence, especially in the reconstruction condition with lowered temperature where the extractable entropy increases from 0.18 using a single measurement to 0.24 using the maximum number of measurements.

The DFF PUF performed slightly better with an estimated min-entropy of 0.5 bits per cell and a compression ratio of 0.86. The modified Daugman method estimated the entropy per cell to be around 0.40 using a single measurement and around 0.43 using multiple measurements, showing a very minor increase of entropy again due to a skewed bias distribution. The extractable entropy of the DFF PUF is 0.33 bit per cell in the worst case using a single measurement during enrollment and reconstruction and 0.46 bit per cell in the optimal reconstruction condition using multiple measurements. This increase is less as seen with the SRAM and buskeeper PUF and the extractable entropy is generally lower, but still better than the latch PUF.

8.3.3 Lessons learned

We found that there are several important aspects contributing to the amount of information that can be extracted from a PUF. First, there is the distribution of the biases of PUF cells (that is, the bias distribution averaged over all PUFs). The analysis shows that the entropy depends on the shape and skewness of this distribution. When a cell prefers to start in a certain state, i.e. there is a certain biasing effecting all PUFs, the extractable entropy decreases. This is a direct consequence of the fact that entropy is only maximal when all possible events are equiprobable.

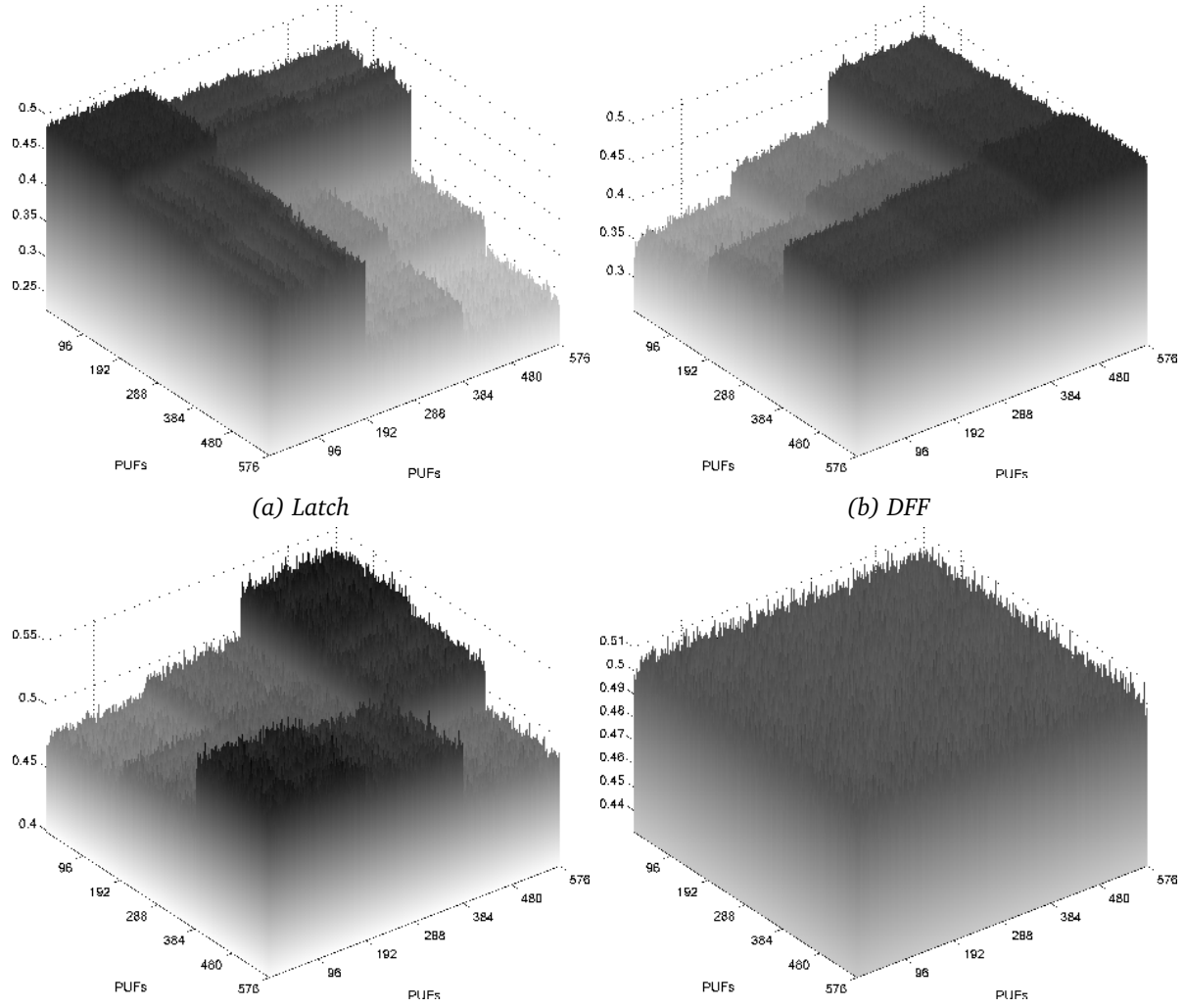
A very important factor here is the number of measurements performed during enrollment and reconstruction. If only one measurement is performed during both stages, the optimal bias

distribution (resulting in the largest entropy) is a uniform distribution with two outcomes, a bias of zero and one. When more measurements are performed, a uniform distribution with more outcomes yields a higher extractable entropy. However, the estimated bias distributions of the cells of all PUF types analysed in this thesis do not follow a uniform distribution using more measurements, but instead have a preference to start almost all measurements as zero or one.

Current PUF usage and processing with fuzzy extractors is usually optimised for a single measurement during enrollment and reconstruction. For this scenario, the estimated bias distribution of the PUFs analysed in this thesis are almost optimal. For example, the estimated bias distribution of the SRAM PUF (Figure 7.2) show that almost all probability lies near the edges. This is the reason that the extractable entropy of the analysed PUFs (only) increases with diminishing effectiveness as the number of measurements used during enrollment and reconstruction increases.

The second very important aspect — also directly influencing the effectiveness of multiple measurements — is the change in bias distribution under reconstruction conditions (noise). The entropy analyses shows that when this change is minimal, as with the SRAM PUF, the extractable entropy does increase (although, as argued in the previous paragraph, this is limited) but that for PUFs with a significant change in bias distribution the effect of additional measurements on the extractable entropy is extremely limited (compare Figures 8.10a-8.10d).

Lastly, possible correlations between cells can have a negative effect on the extractable entropy. However, a Pearson correlation test showed that there are no more significant correlations than would be detected by chance. This test was performed for the synthetic data as well as actual PUF data, finding similar results. This indicates that linear correlations among cells are low or non-existent. Still, the results of the Daugman method, mainly the decrease in the degrees of freedom, indicate that there is some correlation present. In addition, for some PUF types the CTW compression test also hinted at correlation. To determine if there indeed is any correlation present and how large the effect on the extractable entropy is, more tests are needed.

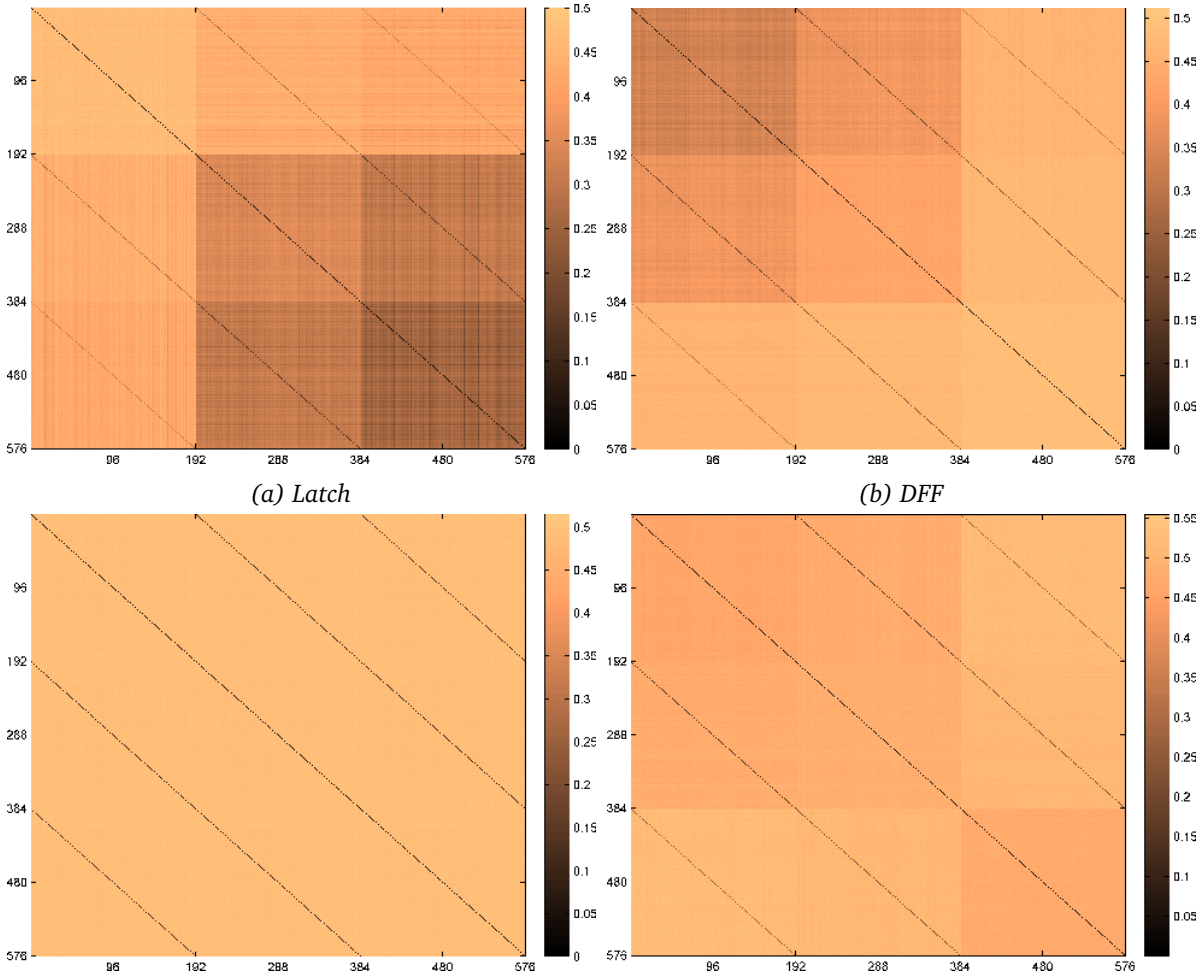


(a) Latch *(b) DFF*

(c) Buskeeper *(d) SRAM*

Figure 8.5: This figure illustrates the inter-device distance per PUF type. Both the horizontal as vertical axis contain devices numbers and the height values denotes the average distance in bias as given in Equation 8.3. Note that the scale of the z-axis (height) is dependent on the PUF type.
The device number denotes 3 times the same 192 devices, where the following table maps a PUF to the condition under which it was measured:

1–192	$-40°C$
193–384	$+25°C$
385–576	$+85°C$

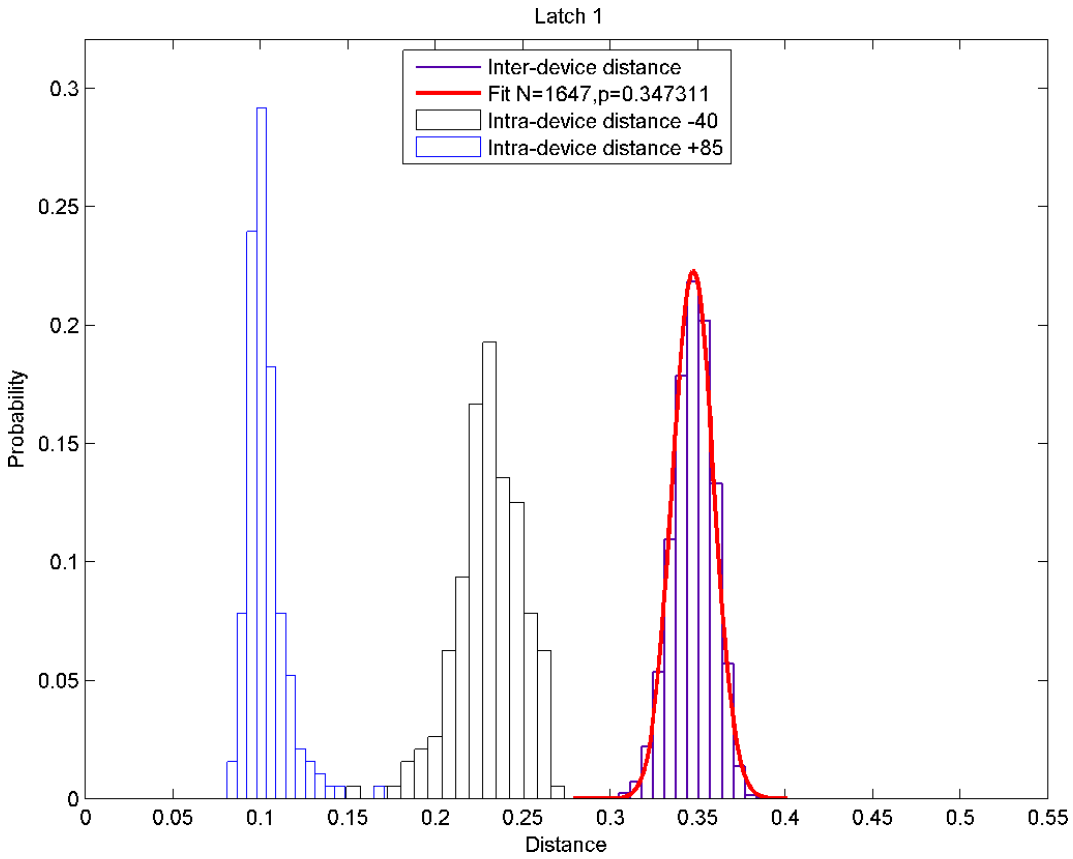


(a) Latch *(b) DFF*

(c) SRAM *(d) Buskeeper*

Figure 8.6: This figure illustrates the inter-device distance per PUF type. Both the horizontal as vertical axis contain devices numbers and the colour values denotes the average distance in bias as given in Equation 8.3. The device number denotes 3 times the same 192 devices, where the following table maps a PUF to the condition under which it was measured:

1–192	$-40°C$
193–384	$+25°C$
385–576	$+85°C$



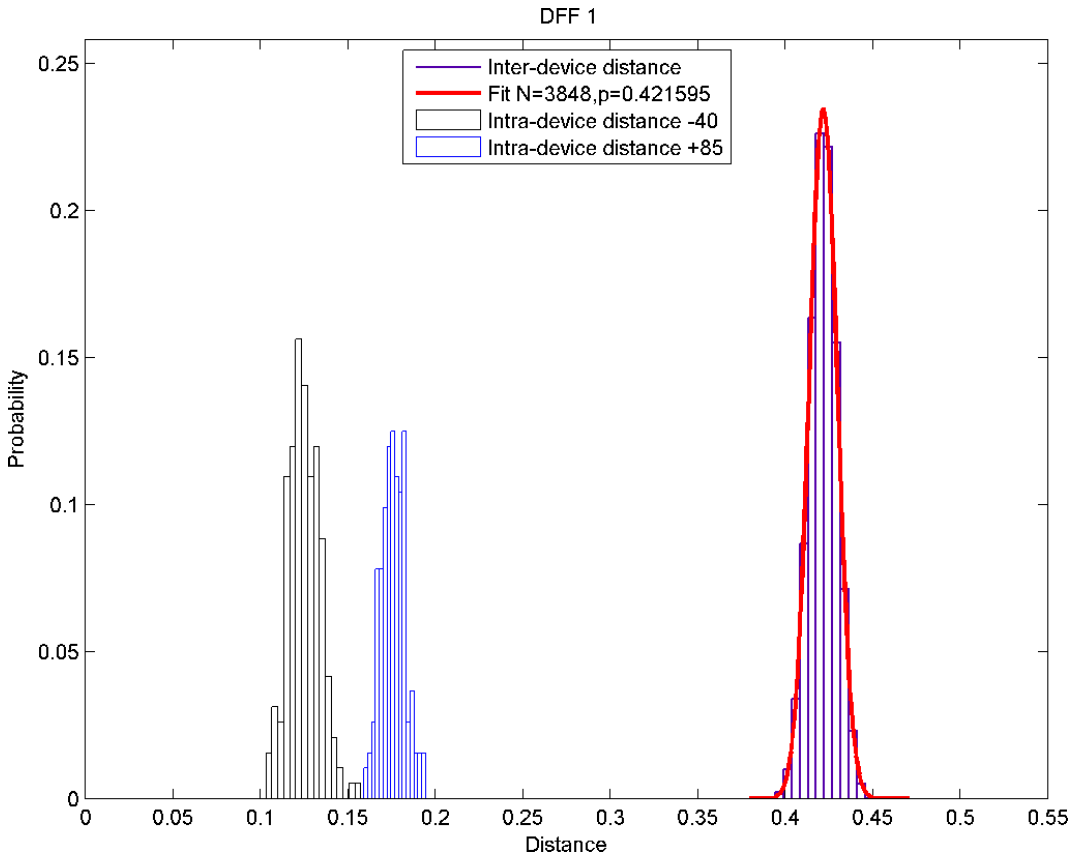


Figure 8.7: Latch and DFF

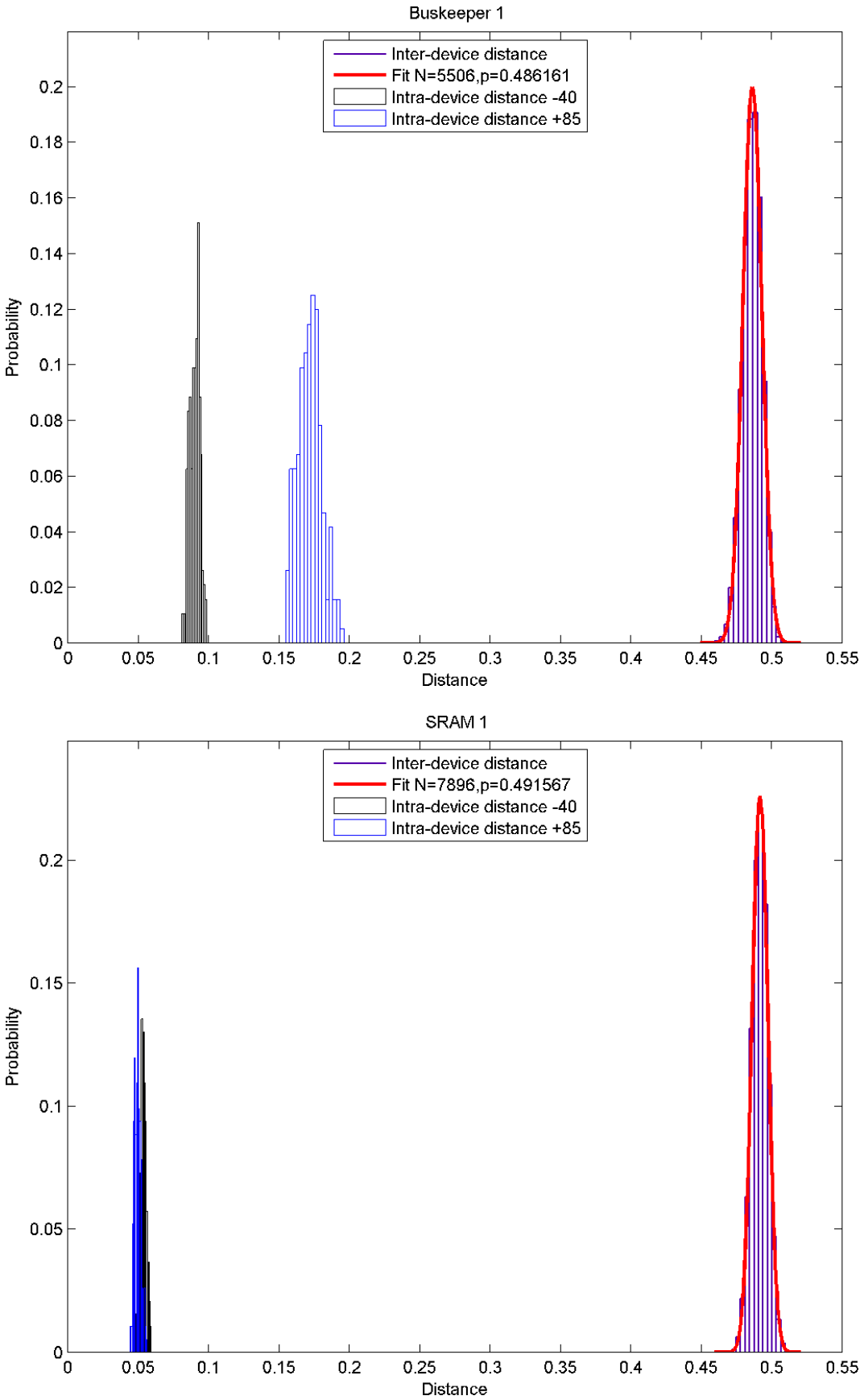


Figure 8.7: Buskeeper and SRAM

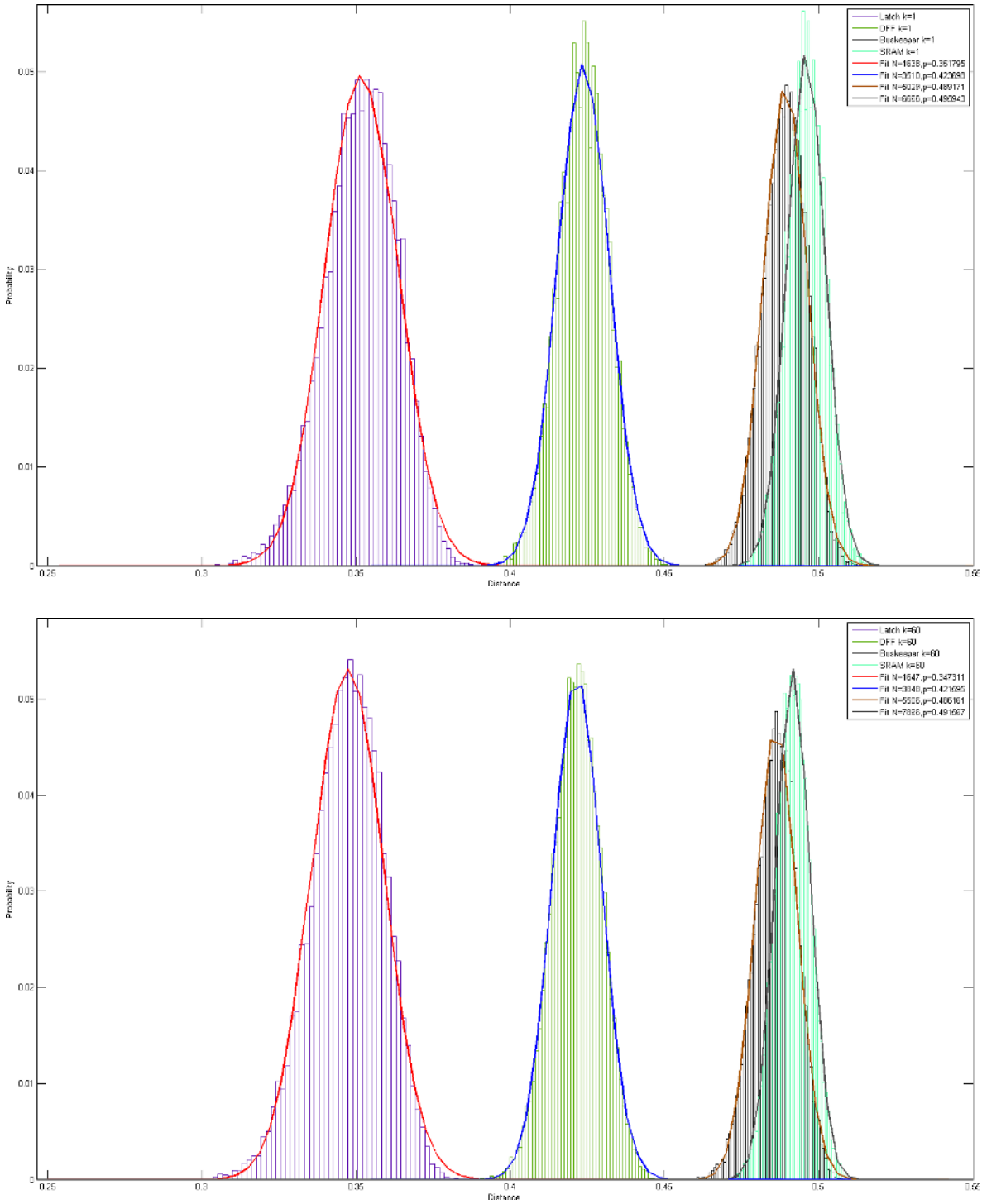


Figure 8.8: The estimated probability distributions of the inter-device distances for the four PUF types. The x-axis denotes the bias distance and the y-axis denotes the probability of having that distance.

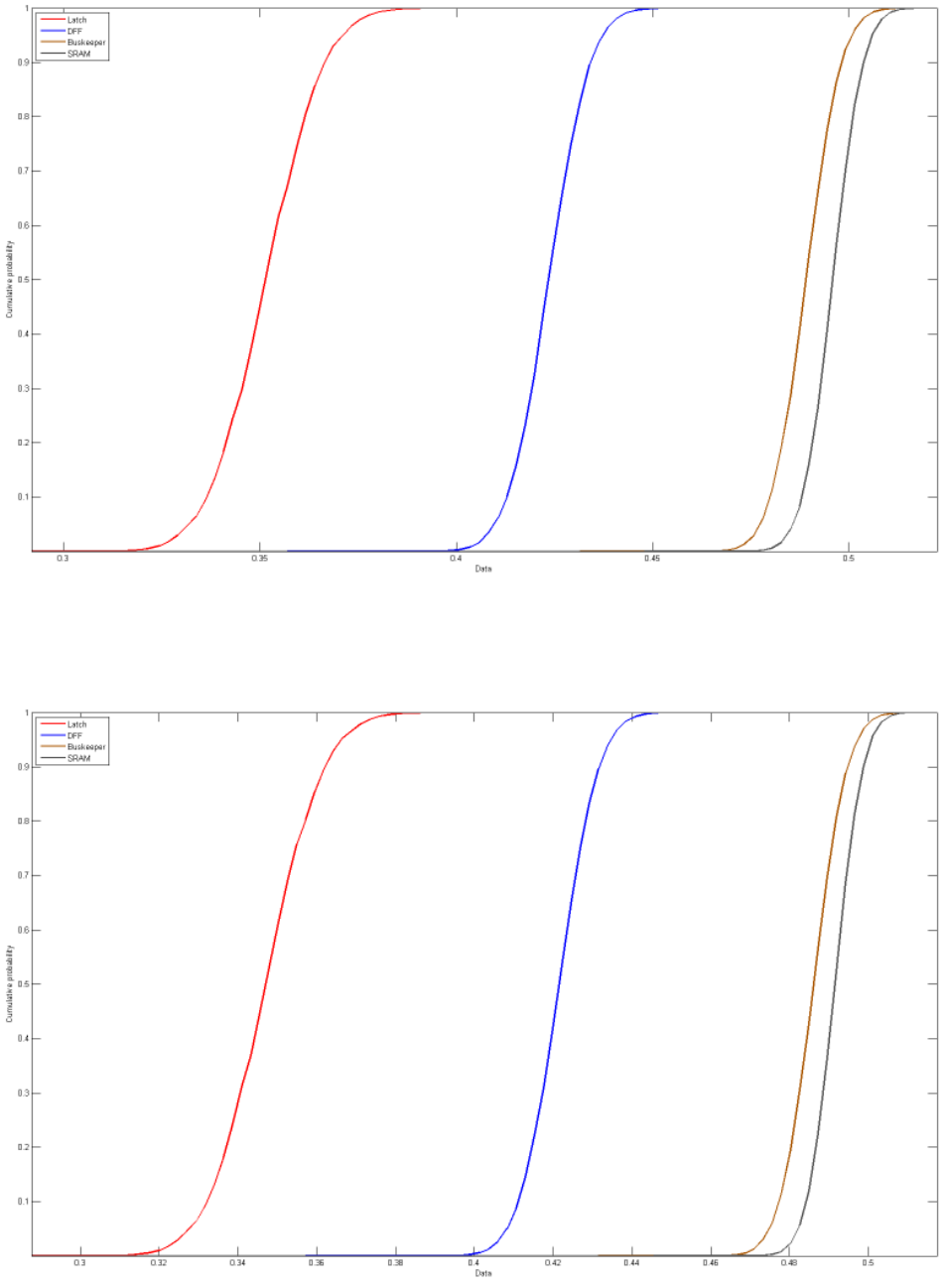
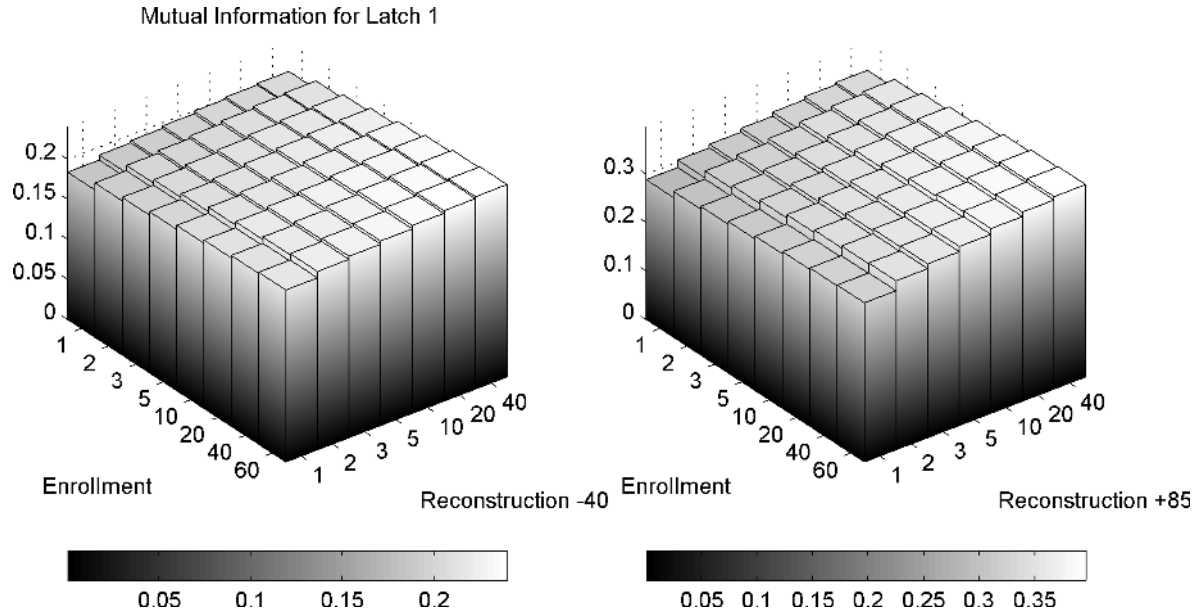
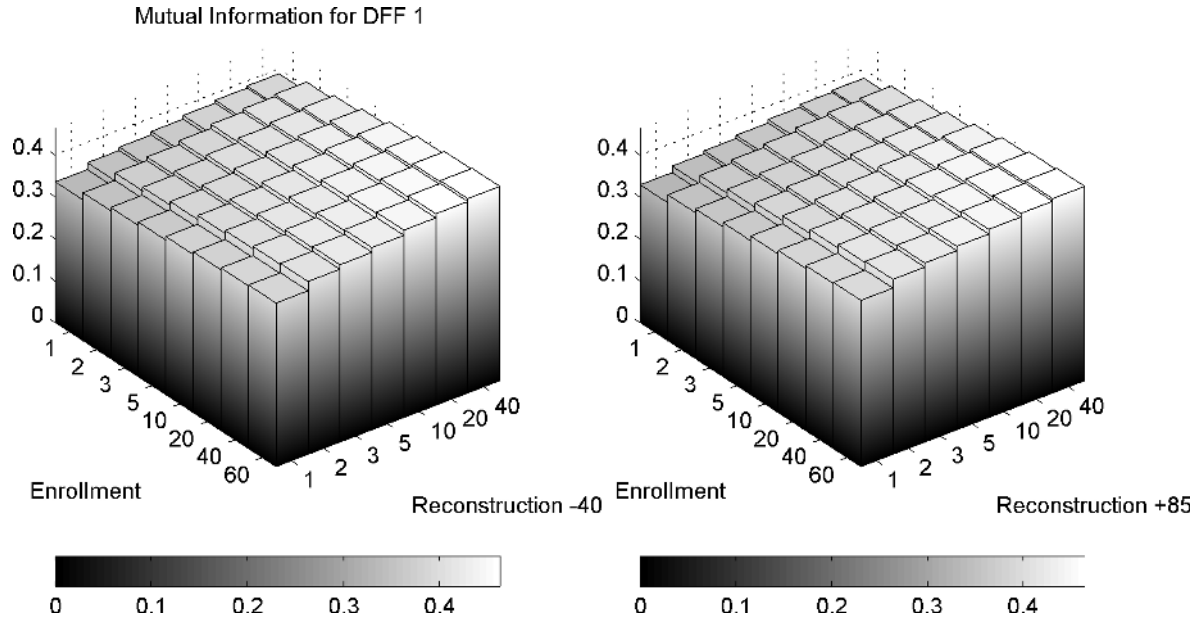


Figure 8.9: Cumulative distribution function for k=1 and k=60 of the four PDFs given in Figure 8.8.

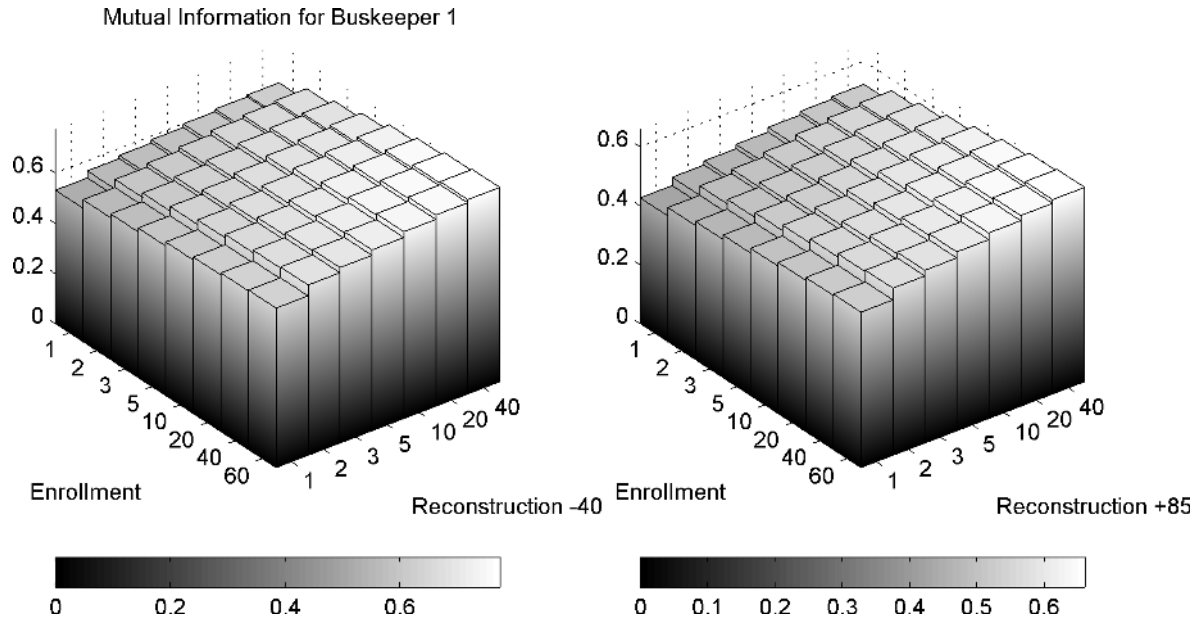


(a) Latch

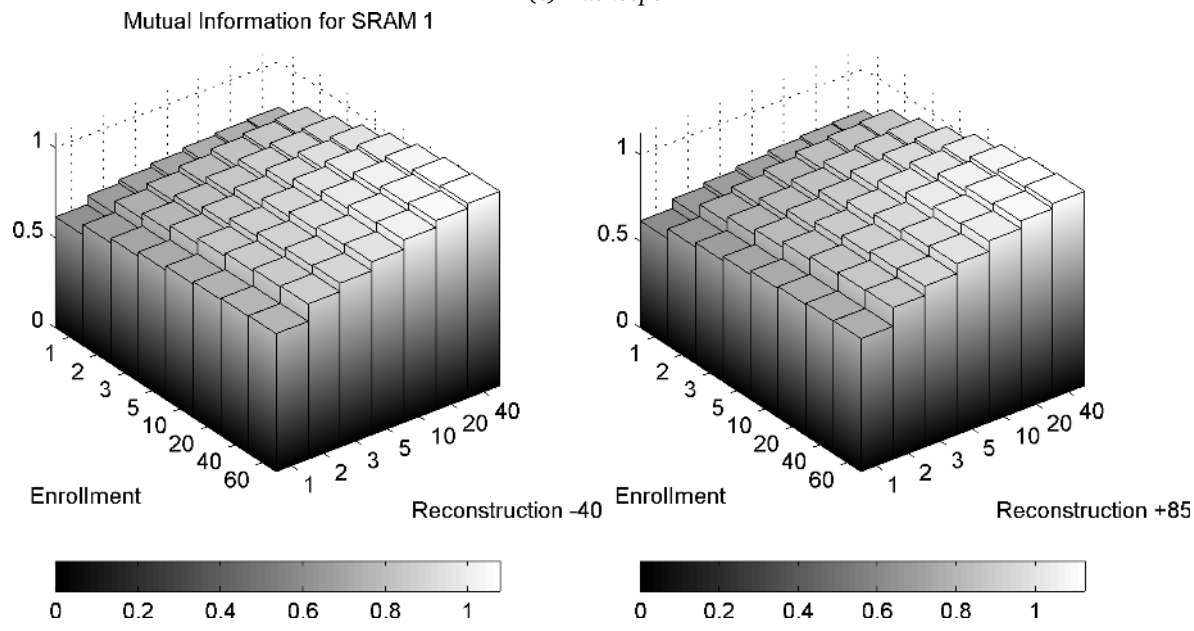


(b) DFF

Figure 8.10: Mutual information of Latch (top) and DFF



(c) Buskeeper



(d) SRAM

Figure 8.10: Mutual information of Buskeeper (top) and SRAM

Chapter 9

Conclusion

The central question of this thesis was to analyse the entropy of physical unclonable functions, which consisted of a theoretical part in which methods to estimate entropy are devised as well as a practical part in which an entropy analysis was performed on PUF data from the UNIQUE project. Here, this thesis is concluded by providing a summary, an overview of the contributions and results, and lastly with a section on limitations and future work.

9.1 Summary

This thesis started with a literature study to investigate the PUF types present on the UNIQUE ASIC as well as the current methodologies to assess uniqueness and estimate entropy. The results of these methods were collected from various works and summarized to serve as references for our own entropy estimation.

We introduced a general framework modelling the PUF creation and processing procedures, quantifying the relevant stochastic variables, denoting the important entropies and providing a clear and concise notation used throughout the thesis for entropy estimation methods. Using this model, cell biases are considered as a discriminative feature rather than bit instability.

After this, the question how the bias distribution of PUF cells can be determined was investigated. A method was proposed that given a limited sample of PUF data estimates an upper bound on the bias probability distribution using the maximum entropy principle.

Furthermore, we introduced two methods to estimate entropy using an arbitrary number of measurements during enrollment and reconstruction. The first method determines the extractable entropy of PUFs by calculating the mutual information between enrollment and reconstruction measurements using the estimated bias probability distributions. By choosing the most detrimental reconstruction condition, a guarantee can be given on the minimal entropy that can be extracted from a PUF when it is used in practice.

In addition, a method used for iris recognition was modified in such a way that it is suitable to estimate the creation entropy of PUFs. This method estimates the entropy present in enrollment responses based on the degrees of freedom in the distribution of inter-device bias distances.

Synthetic PUF data was generated to analyse the performance of entropy estimation methods and to investigate the influence on the analysis of various factors such as the number of PUFs used and the number of measurements used during enrollment and reconstruction.

Lastly, an entropy analysis was performed on the four memory based PUF types present on the UNIQUE ASIC. To this end, various Matlab scripts have been written to pre-process the UNIQUE

data, apply the entropy estimation methods and to create the figures and tables displayed in this thesis. The results were presented and compared to those in the literature.

9.2 Results

The distribution of cell biases of four memory based PUF types were estimated using a large dataset from the UNIQUE project. We found that all PUF types have biases concentrated near zero or one, meaning that it is uncommon for a cell to start-up in different states over multiple measurements.

This also proved that our proposed method to estimate an upper bound on the bias probability distribution of cells was unsuitable for the PUF data since it provides a Gaussian estimate while histograms of the PUF data show that the probability of a cells' bias is not at all Gaussian distributed. Hence, it is better to base the estimate of the bias distribution on histograms of the empirical data.

Furthermore, we investigated the influence of the number of measurements used during enrollment and reconstruction on the extractable entropy. Due to the shape of the bias distribution, this increase is not very large. However PUFs that are very reliable under reconstruction conditions benefit from multiple measurements and can extract almost a factor two more entropy than with a single measurement during enrollment and reconstruction.

Furthermore, we investigated the correlation among cell biases, and found that none of the PUF types investigated have a linear correlation between the start-up values of cells according to the Pearson correlation test. However, our modified Daugman method suggest that there may be correlations as the degrees of freedom is significantly reduced in the real PUF data when compared to synthetic data which is known to be independent.

From the analysis of the synthetic PUF data, we found that results for the entropy estimation methods strongly differ. The CTW compression test seems to be a good predictor for creation entropy in case a single measurement is used while the min-entropy provides an under-estimation. The prediction for the extractable entropy using our mutual information method appears to be quite accurate.

The number of PUFs required to accurately estimate entropy depends on the test used. Around 20 samples should suffice for the CTW compression test, while the min-entropy and extractable entropy method using a global noise model should have at least 80 samples. For the modified Daugman method and the extractable entropy test when using an individual noise model per cell, even more samples are required.

When applying our methods to PUF data from the UNIQUE project, we found that the results are in line with previous work. Both our methods show that the SRAM PUF has the strongest PUF behaviour of the four tested PUF types. The extractable entropy of this PUF type lies between 0.61 and 1.12. This depends on the number of measurements used during enrollment and reconstruction as the extractable information increases when more measurements are used, but only if the reliability between enrollment and reconstruction conditions is good.

9.3 Limitations and future work

In the section on the estimation of the bias probability distribution, we only considered estimating this probability distribution for a single PUF cell at a time. Future work could extend this by considering bias distribution estimation in the multivariate case. The same holds for our approach

to calculate the extractable entropy, which is also limited in the sense that PUF cells are assumed to be independent. Future work could generalize this approach for the multivariate case and even include correlations. Further future work regarding the extractable entropy method is to analyse an even larger dataset or reduce the number of bins used during enrollment and reconstruction, such that there is no need for a global noise model. In this manner, a more accurate entropy estimation can be obtained. Furthermore, the method we proposed to estimate an upper bound on the bias probability distribution by applying the maximal entropy principle turned out to be not applicable for our PUF data.

Appendix A

Proof of maximal entropy bias distribution

Here, we consider the univariate case, thus estimating the probability distribution of a single cell. As per the maximum entropy principle, we wish to determine $\rho_1(b)$ by maximising the differential entropy formula

$$h(B) = -\int_0^1 \mathrm{d}b\, \rho_1(b) \ln \rho_1(b) \tag{A.1}$$

with

1. $\rho_1(b) \geq 0$
2. $\int_0^1 \rho_1(b)\mathrm{d}b = 1$
3. $\int_0^1 \mathrm{d}b\, \rho_1(b) \cdot b = \mu$
4. $\int_0^1 \mathrm{d}b\, \rho_1(b) \cdot b^2 = \mu^2 + \sigma^2$

as constraints. To do this, we form the Lagrangian

$$\Lambda(\rho_1, \lambda_0, \lambda_1, \lambda_2) = h(B) + \lambda_0\Big(\int_0^1 \mathrm{d}b\, \rho_1(b) - 1\Big) + \lambda_1\Big(\int_0^1 \mathrm{d}b\, \rho_1(b)\cdot b - \mu\Big) + \lambda_2\Big(\int_0^1 \mathrm{d}b\, \rho_1(b)\cdot b^2 - (\mu^2+\sigma^2)\Big) \tag{A.2}$$

in which we maximise $h(B)$ by differentiating with respect to $\rho_1(b)$ and setting the result equal to zero:

$$
\begin{aligned}
0 = \frac{\delta\Lambda}{\delta\rho_1(b_*)} &= -\int_0^1 \mathrm{d}b \left[\ln\rho_1(b)\delta(b-b_*) + \rho_1(b)\frac{1}{\rho_1(b)}\delta(b-b_*)\right] \\
&\quad + \lambda_0\int_0^1 \mathrm{d}b\,\delta(b-b_*) + \lambda_1\int_0^1 \mathrm{d}b\,b\delta(b-b_*) \\
&\quad + \lambda_2\int_0^1 \mathrm{d}b\,b^2\delta(b-b_*) \\
&= -\ln\rho_1(b_*) - 1 + \lambda_0 + \lambda_1 b_* - \lambda_2 b_*^2 \\
\rho_1(b_*) &= e^{-1+\lambda_0+b_*\lambda_1-b_*^2\lambda_2} \\
\rho_1(b) &\propto e^{-\lambda_2 b^2+\lambda_1 b}.
\end{aligned}
\tag{A.3}
$$

Now, λ_1, λ_2 have to be chosen such that all constraints are satisfied. Starting with constraint 2:

$$
N = \int_0^1 \mathrm{d}b\, e^{-\lambda_2 b^2+\lambda_1 b} = \left.\frac{e^{\frac{\lambda_1^2}{4\lambda_2}}\sqrt{\pi}\,\mathrm{Erf}(\frac{2\lambda_2 b-\lambda_1}{2\sqrt{\lambda_2}})}{2\sqrt{\lambda_2}}\right|_0^1 = \frac{\sqrt{\pi}e^{\frac{\lambda_1^2}{4\lambda_2}}\left(\mathrm{Erf}\left(\frac{\lambda_1}{2\sqrt{\lambda_2}}\right)+\mathrm{Erf}\left(\frac{2\lambda_2-\lambda_1}{2\sqrt{\lambda_2}}\right)\right)}{2\sqrt{\lambda_2}}.
$$

Dividing the old ρ_1 by N results in a new probability density function satisfying constraint 2:

$$
\rho_1(b) = -\frac{2\sqrt{\lambda_2}e^{-\lambda_2 b^2+\lambda_1 b-\lambda_1^2/(4\lambda_2)}}{\sqrt{\pi}\left(\mathrm{Erf}\left(\frac{\lambda_1}{2\sqrt{\lambda_2}}\right)+\mathrm{Erf}\left(\frac{2\lambda_2-\lambda_1}{2\sqrt{\lambda_2}}\right)\right)}.
\tag{A.4}
$$

Here, the relation between μ,σ^2 and λ_1, λ_2 is of interest. The mean and variance can be extracted from the data to obtain the parameters for the bias distribution. To find these, we need to consider the third and fourth constraint.

The third constraint of the mean is defined as $\mu = \int_0^1 b\cdot\rho_1(b)\mathrm{d}b$. Solving this using a symbolic computing environment, gives, after simplifying:

$$
\mu = \frac{\lambda_1}{2\lambda_2} + \frac{e^{-c^2}-e^{-d^2}}{\sqrt{\lambda_2\pi}(\mathrm{Erf}[c]+\mathrm{Erf}[d])}
\tag{A.5}
$$

where

$$
\begin{aligned}
c &= \frac{\lambda_1}{2\sqrt{\lambda_2}},\ c^2 = \frac{\lambda_1^2}{4\lambda_2}, \\
d &= \frac{2\lambda_2-\lambda_1}{2\sqrt{\lambda_2}},\ d^2 = c^2+\lambda_2-\lambda_1.
\end{aligned}
$$

Constraint four is used in a similar way to obtain the variance, which can be calculated as $\sigma^2 = \int_0^1 b^2\rho_1(b)\mathrm{d}b - \mu^2$. Again, solving this using a symbolic computing environment gives:

$$
\sigma^2 = \frac{\lambda_1\mu+1}{2\lambda_2} - \mu^2 - \frac{e^{-d^2}}{\sqrt{\lambda_2\pi}(\mathrm{Erf}[c]+\mathrm{Erf}[d])}
\tag{A.6}
$$

The parameters λ_1 and λ_2 of this maximum entropy distribution can be found by reversing these formulas, that is, writing λ_1 and λ_2 as a function of μ and σ^2. Then, by filling in these parameters, the bias probability density function is known.

Bibliography

[1] *Project UNIQUE website*, `http://www.unique-project.eu/`.

[2] Kanak Agarwal and Sani Nassif, *The impact of random device variation on SRAM cell stability in sub-90nm CMOS technologies*, IEEE Trans. Very Large Scale Integr. Syst. **16** (2008), no. 1, 86–97.

[3] N.A. Ahmed and D.V. Gokhale, *Entropy expressions and their estimators for multivariate distributions*, IEEE Transactions on Information Theory **35** (1989), no. 3, 688–692.

[4] Frederik Armknecht, Roel Maes, Ahmad-Reza Sadeghi, Franccois-Xavier Standaert, and Christian Wachsmann, *A Formal Foundation for the Security Features of Physical Functions*, IEEE Security & Privacy (2011), no. 1, 16.

[5] Hagai Bar-El, *Known attacks against smartcards*, White paper, available online at: `http://www.hbarel.com/publications/Known_Attacks_Against_Smartcards.pdf`.

[6] A.J. Bhavnagarwala, Xinghai Tang, and J.D. Meindl, *The impact of intrinsic device fluctuations on CMOS SRAM cell stability*, IEEE Journal of Solid-state Circuits **36** (2001), 658–665.

[7] Heike Busch, Miroslava Sotáková, Stefan Katzenbeisser, and Radu Sion, *The PUF promise*, Trust and Trustworthy Computing (Alessandro Acquisti, Sean Smith, and Ahmad-Reza Sadeghi, eds.), Lecture Notes in Computer Science, vol. 6101, Springer Berlin / Heidelberg, 2010, pp. 290–297.

[8] Mathias Claes, Vincent van der Leest, and An Braeken, *Comparison of SRAM and FF PUF in 65nm technology*, Information Security Technology for Applications (Peeter Laud, ed.), Lecture Notes in Computer Science, vol. 7161, Springer Berlin / Heidelberg, 2012, pp. 47–64.

[9] Thomas M. Cover and Joy A. Thomas, *Elements of information theory (Wiley series in telecommunications and signal processing)*, Wiley-Interscience, 2006.

[10] John Daugman, *The importance of being random: statistical principles of iris recognition*, Pattern Recognition **36** (2003), 279–291.

[11] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith, *Fuzzy extractors: How to generate strong keys from biometrics and other noisy data*, SIAM J. Comput. **38** (2008), no. 1, 97–139.

[12] Blaise Gassend, *Physical Random Functions*, Master's thesis, MIT, USA, 2003.

[13] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas, *Controlled physical random functions*, ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference (Washington, DC, USA), IEEE Computer Society, 2002, p. 149.

[14] Blaise Gassend, Marten Van Dijk, Dwaine Clarke, Emina Torlak, Srinivas Devadas, and Pim Tuyls, *Controlled physical random functions and applications*, ACM Trans. Inf. Syst. Secur. **10** (2008), no. 4, 1–22.

[15] Blaise Gassend, Daihyun Lim, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas, *Identification and authentication of integrated circuits: Research articles*, Concurr. Comput. : Pract. Exper. **16** (2004), no. 11, 1077–1098.

[16] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls, *FPGA intrinsic PUFs and their use for IP protection*, Cryptographic Hardware and Embedded Systems Workshop, LNCS, vol. 4727.

[17] _____, *Physical unclonable functions, FPGAs and public-key crypto for IP protection*, FPL (Koen Bertels, Walid A. Najjar, Arjan J. van Genderen, and Stamatis Vassiliadis, eds.), IEEE, 2007, pp. 189–195.

[18] Jorge Guajardo, Boris Škorić, Pim Tuyls, Sandeep S. Kumar, Thijs Bel, Antoon H. Blom, and Geert-Jan Schrijen, *Anti-counterfeiting, key distribution, and key storage in an ambient world via physical unclonable functions*, Information Systems Frontiers **11** (2009), no. 1, 19–41.

[19] Helena Handschuh, Geert-Jan Schrijen, and Pim Tuyls, *Hardware intrinsic security from physically unclonable functions*, Towards Hardware-Intrinsic Security (Ahmad-Reza Sadeghi and David Naccache, eds.), Information Security and Cryptography, Springer Berlin Heidelberg.

[20] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu, *Initial SRAM state as a fingerprint and source of true random numbers for RFID tags*, Proceedings of the Conference on RFID Security, July 2007.

[21] Gabriel Hospodar, Roel Maes, and Ingrid Verbauwhede, *Implications of Machine Learning Attacks on Arbiter PUF-based Challenge-Response Authentication and Secure Key Generation*, Cosic internal report, 2012.

[22] Tanya Ignatenko, Geert-Jan Schrijen, Boris Škorić, Pim Tuyls, and Frans M. J. Willems, *Estimating the secrecy rate of physical uncloneable functions with the context-tree weighting method*, Proc. IEEE International Symposium on Information Theory 2006 (Seattle, USA), July 2006, pp. 499–503.

[23] D. Karakoyunlu and B. Sunar, *Differential template attacks on PUF enabled cryptographic devices*, Information Forensics and Security (WIFS), 2010 IEEE International Workshop on, dec. 2010, pp. 1 –6.

[24] Stefan Katzenbeisser, Ünal Kočabas, Vincent van der Leest, Ahmad-Reza Sadeghi, Geert-Jan Schrijen, Heike Schröder, and Christian Wachsmann, *Recyclable PUFs: Logically reconfigurable PUFs*, Cryptographic Hardware and Embedded Systems CHES 2011 (Bart Preneel and Tsuyoshi Takagi, eds.), Lecture Notes in Computer Science, vol. 6917, Springer Berlin / Heidelberg, 2011, pp. 374–389.

[25] Patrick Koeberl, Roel Maes, Vladimir Rožić, Vincent Van der Leest, Erik Van der Sluis, and Ingrid Verbauwhede, *Experimental evaluation of physically unclonable functions in 65 nm CMOS*, 38th European Solid-State Circuits Conference (ESSCIRC 2012), IEEE, 2012.

[26] Sandeep S. Kumar, Jorge Guajardo, Roel Maes, Geert-Jan Schrijen, and Pim Tuyls, *Extended abstract: The butterfly PUF protecting IP on every FPGA*, Proceedings of the 2008 IEEE International Workshop on Hardware-Oriented Security and Trust (Washington, DC, USA), IEEE Computer Society, 2008, pp. 67–70.

[27] Klaus Kursawe, Ahmad-Reza Sadeghi, Dries Schellekens, Pim Tuyls, and Boris Škorić, *Reconfigurable physical unclonable functions – enabling technology for tamper-resistant storage*, 2nd IEEE International Workshop on Hardware-Oriented Security and Trust - HOST 2009 (San Francisco,CA,USA), IEEE, 2009, pp. 22–29.

[28] Jae W. Lee, Daihyun Lim, Blaise Gassend, G. Edward Suh, Marten Van Dijk, and Srini Devadas, *A technique to build a secret key in integrated circuits with identification and authentication applications*, In Proceedings of the IEEE VLSI Circuits Symposium, 2004, pp. 176–179.

[29] Daihyun Lim, *Extracting secret keys from integrated circuits*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems (2004).

[30] Roel Maes, Pim Tuyls, and Ingrid Verbauwhede, *Intrinsic PUFs from flip-flops on reconfigurable devices*, 3rd Benelux Workshop on Information and System Security (WISSec 2008) (Eindhoven,NL), 2008, p. 17.

[31] Roel Maes and Ingrid Verbauwhede, *Physically unclonable functions: A study on the state of the art and future research directions*, Towards Hardware-Intrinsic Security (Ahmad-Reza Sadeghi and David Naccache, eds.), Information Security and Cryptography, Springer Berlin Heidelberg, 2010, pp. 3–37.

[32] Abhranil Maiti, Vikash Gunreddy, and Patrick Schaumont, *A systematic method to evaluate and compare the performance of physical unclonable functions*, IACR Cryptology ePrint Archive **2011** (2011), 657.

[33] Dominik Merli, Dieter Schuster, Frederic Stumpf, and Georg Sigl, *Semi-invasive EM attack on FPGA RO PUFs and countermeasures*, Proceedings of the Workshop on Embedded Systems Security (New York, NY, USA), WESS '11, ACM, 2011, pp. 2:1–2:9.

[34] Ravikanth S. Pappu, *Physical one-way functions*, Ph.D. thesis, Massachusetts Institute of Technology, March 2001.

[35] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber, *Modeling attacks on physical unclonable functions*, CCS 2010: Proceedings of the 17th ACM conference on Computer and communications security (New York, NY, USA), ACM, 2010, pp. 237–249.

[36] Geert-Jan Schrijen and Vincent van der Leest, *Comparative analysis of SRAM memories used as PUF primitives*, Design, Automation Test in Europe Conference Exhibition (DATE), 2012, march 2012, pp. 1319 –1324.

[37] Dieter K. Schroder, *Negative bias temperature instability: What do we understand?*, Microelectronics Reliability **47** (2007), no. 6, 841 – 852.

[38] Dieter Schuster, *Side-channel analysis of physical unclonable functions (PUFs)*, Diploma thesis, Technische Universität München, December 2010.

[39] Georgios N. Selimis, Mario Konijnenburg, Maryam Ashouei, Jos Huisken, Harmke de Groot, Vincent van der Leest, Geert Jan Schrijen, Marten van Hulst, and Pim Tuyls, *Evaluation of 90nm 6T-SRAM as physical unclonable function for secure key generation in wireless sensor nodes.*, ISCAS, IEEE, 2011, pp. 567–570.

[40] C. E. Shannon, *A mathematical theory of communication*, Bell system technical journal **27** (1948).

[41] Peter Simons, Erik van der Sluis, and Vincent van der Leest, *Buskeeper PUFs, a promising alternative to D flip-flop PUFs*, IEEE Int. Symposium on Hardware-Oriented Security and Trust.

[42] Ying Su, Jeremy Holleman, and Brian P. Otis, *A Digital 1.6 pJ/bit Chip Identification Circuit Using Process Variations*, IEEE Journal of Solid-State Circuits **43** (2008), no. 1, 69–77.

[43] G. Edward Suh and Srinivas Devadas, *Physical unclonable functions for device authentication and secret key generation*, Design Automation Conference (New York, NY, USA), ACM Press, 2007, pp. 9–14.

[44] P. Tuyls, B. Škorić, S. Stallinga, T. Akkermans, and W. Ophey, *An information theoretic model for physical uncloneable functions*, Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on, June-2 July 2004, pp. 141–.

[45] Pim Tuyls, Geert-Jan Schrijen, Boris Škorić, Jan van Geloven, Nynke Verhaegh, and Rob Wolters, *Read-proof hardware from protective coatings*, Cryptographic Hardware and Embedded Systems Workshop, LNCS, vol. 4249, Springer, October 2006, pp. 369–383.

[46] Pim Tuyls and Boris Škorić, *Strong authentication with physical unclonable functions*, Security, Privacy, and Trust in Modern Data Management (Milan Petkovic and Willem Jonker, eds.), Data-Centric Systems and Applications, Springer Berlin Heidelberg, 2007, pp. 133–148.

[47] Vincent van der Leest, *UNIQUE newsletter October 2011*, Newsletter, available online at: `http://www.unique-project.eu/downloads/UNIQUE-238811-Newsletter-October-2011.pdf`.

[48] Vincent van der Leest, Erik van der Sluis, Geert-Jan Schrijen, Pim Tuyls, and Helena Handschuh, *Cryptography and security*, 2012.

[49] Ingrid Verbauwhede and Roel Maes, *Physically unclonable functions: manufacturing variability as an unclonable device identifier*, Proceedings of the 21st edition of the great lakes symposium on Great lakes symposium on VLSI (New York, NY, USA), GLSVLSI '11, ACM, 2011, pp. 455–460.

[50] B. Škorić, *Physical aspects of digital security*, 2011, Lecture notes.

[51] B. Škorić, S. Maubach, T. Kevenaar, and P. Tuyls, *Information-theoretic analysis of capacitive physical unclonable functions*, Journal of Applied Physics **100** (2006), no. 2, 024902.

[52] Frans M.J. Willems, Yuri M. Shtarkov, and Tjalling J. Tjalkens, *The context tree weighting method: Basic properties*, IEEE Transactions on Information Theory **41** (1995), 653–664.